

ESTIMATION AND OPTIMIZATION OF FEATURE SELECTION STABILITY WITH APPLICATIONS TO OMICS DATA

Victor Hamer

*Thesis submitted in partial fulfillment of the requirements for
the Degree of Doctor in Applied Sciences*

November 2022

ICTEAM
Louvain School of Engineering
UCLouvain
Louvain-la-Neuve
Belgium

Thesis Committee:

Pr. Pierre Dupont , <i>Supervisor</i>	UCLouvain/ICTEAM, Belgium
Pr. Michel Verleysen , <i>President</i>	UCLouvain/ICTEAM, Belgium
Pr. Benjamin Haibe-Kains	University of Toronto/MBP, Canada
Pr. Charles Pecheur , <i>Secretary</i>	UCLouvain/ICTEAM, Belgium
Pr. Yvan Saey s	Ghent University, Belgium
Dr. Cyril De Bodt	UCLouvain/ICTEAM, Belgium

Estimation and optimization of feature selection stability with applications to omics data

by Victor Hamer

© Victor Hamer 2022
ICTEAM
UCLouvain
Place Sainte-Barbe, 2
1348 Louvain-la-Neuve
Belgium

This work was partially supported by the F.R.S-FNRS.

Preamble

Machine learning’s impact on society is rapidly increasing. Machine learning has been successfully applied to numerous daily life problems, from virtual assistants to traffic prediction. Research is being done on more and more sensitive prospects, such as the development of self-driving cars or diagnosis and prognosis kits in clinics. Such kits are supposed to help doctors to make adequate decisions. For instance, a prognosis tool discriminating between breast-cancer patients with a low or high risk of cancer relapse can assist oncologists in the prescribed treatment choice. A competitive prognostic kit could save lives while sparing some low-risk patients from the discomfort of unnecessary chemotherapy. In this context, these doctors’ trust in the machine learning algorithms is primordial. Designing more trustworthy machine learning algorithms is key to promoting machine learning in sensitive areas, such as the biomedical domain. This last statement naturally raises the issue of measuring the degree of trustworthiness of a machine learning algorithm.

This thesis tackles this issue in the specific context of feature selection. Feature selection refers to the area of machine learning which aims to reduce the number of input variables when developing a predictive model. It has become compulsory for a wide variety of applications due to the appearance of very high dimensional data sets, notably in the biomedical domain [SIL07]. For instance, feature selection is an important issue when one deals with large amounts of *omics* data, routinely produced in the context of precision medicine. These data are produced through a vast collection of fastly evolving technologies such as DNA or RNA sequencing. In this instance, the central goal of feature selection is to find a relevant and informative subset of biomarkers from a possibly very large collection of candidate markers. Selecting relevant biomarkers is critical for a better understanding of the underlying biological processes but also for reducing the cost and facilitating the implementation of predictive or prognosis kits in clinics.

Feature selection has already been studied in depth [TAL14; SIL07; KPH07]. Nevertheless, state-of-the-art methods are still somewhat unsatisfactory, mainly because of the typical instability they exhibit. Instability here refers to the fact that the selected markers may drastically change after marginal modifications of the sampling of patients or, more generally, after some fine-tuning of the data analysis pipeline. Instability prevents a sound

and correct interpretation of the selected markers and strongly impacts further biological/clinical validations. Domain experts tend not to trust unstable techniques and often prefer a more stable feature selection algorithm over an unstable and slightly more accurate one [KPH07; SAV08a].

The following two conditions must hold for feature selection algorithms to have a realistic chance of being trusted by domain experts, which is necessary for their broad use in sensitive areas. First, these algorithms must be sufficiently stable according to a specific criterion called a stability measure. Second, this stability measure must adequately quantify the degree of trust from domain experts towards the learning algorithms. This thesis tackles both of these aspects. In particular, its main contributions are the following.

1. *A bi-objective evaluation framework to assess the aggregated quality of feature selection.*

Many papers consider selection stability as a side-objective of prediction performance. Yet, they do not evaluate the aggregated quality of feature selection with a single measure combining both the predictive performance and selection stability. The proposed bi-objective evaluation framework fills this gap and assesses the aggregated quality of Pareto-optimal fronts in the (accuracy, stability) objective space. It aims to quantify the average satisfaction of domain experts with respect to a set of (accuracy, stability) trade-offs. Confidence intervals and hypothesis tests on this metric are constructed and validated on pseudo-real data. These tools can notably assess the statistical significance of observed differences between Pareto-optimal fronts.

2. *The hybrid Recursive Feature Elimination (hybrid-RFE) algorithm: an approach to optimize jointly predictive accuracy and selection stability.*

This approach extends the Recursive Feature Elimination algorithm by enforcing the selection of some features based on a stable, univariate criterion. It allows domain experts to control selection stability via an explicit meta-parameter. They can then choose a particular accuracy-stability trade-off based on their personal preferences. Experiments conducted on six high-dimensional microarray data sets illustrate that large stability gains are obtained with no significant accuracy drop.

3. *The illustration that current state-of-the-art stability measures are not adequate to quantify the trustworthiness of feature selection.*

The inadequacy of current stability measures is illustrated in two scenarios: 1) when jointly optimizing selection stability with predictive performance and 2) when the feature space is composed of highly correlated feature groups. In these scenarios, situations often occur where

the learning algorithm has high stability, according to current measures, but would still be regarded as untrustworthy by domain experts. This claim is supported by theoretical results, simulation studies, and experiments on real microarray and mass-spectrometric data.

4. *A novel importance-weighted stability measure (Φ_{iw}) which accurately estimates the trustworthiness of feature selection.*

The relative importance of the selected features in predictive models is incorporated into this new and highly interpretable measure, which overcomes the limitations of current works. This is illustrated in the same scenarios as the previous contribution: when jointly optimizing selection stability with predictive performance and when the feature space is composed of highly correlated feature groups. The adequacy of the measure is supported by theoretical results, simulation studies, and experiments on real data. The measure also provably satisfies several good properties: it is fully defined, bounded by constants, corrected for chance, and maximal if and only if the importance distribution is deterministic. Side contributions include a generic way to evaluate the importance of a feature in a predictive model and a visualization tool, feature stability maps, allowing the intuitive assessment of stability.

5. *A subsequent novel stability measure (Φ_{msi}) which generalizes Φ_{iw} and considers feature similarities.*

This measure specifically addresses the problem of assessing stability in the presence of strongly correlated features. Correctly measuring selection stability in this context amounts to estimating to which extent several correlated variables contribute to predictive models and how such contributions may change with the data sampling. This novel stability index Φ_{msi} factors in such multivariate contributions. Computing this stability index requires solving a weighted bipartite matching problem to discover which variables actually share these contributions. The measure Φ_{msi} provides more robust stability estimates than current measures, including existing ones taking into account feature correlations. This is demonstrated theoretically, on simulated and real data, including microarray and mass spectrometry data sets.

6. *An interactive visualization tool for domain analysis related to feature selection, under the form of an R package.*

This is primarily a software contribution. The goals of this R package are twofold:

- (a) helping domain experts understand why an algorithm is found to be (un)stable and the implications.

- (b) helping domain experts identify the most interesting features relative to a specific prediction task.

It further allows visualizing intuitively the solution of the weighted bipartite matching problem that is solved to compute Φ_{msi} . This helps identify features not selected with a high frequency but representative of interesting underlying processes. Finally, the package studies the biological processes represented by the selected features, taking advantage of the Gene Ontology.

The thesis is organized as follows. Chapter 1 further sets the context and motivation behind this work. Chapter 2 covers the relevant machine learning background. In particular, it reviews common feature selection algorithms as well as existing stability measures, their evolution, and their properties. Chapter 3 presents the proposed bi-objective evaluation framework and Chapter 4 introduces the hybrid-RFE algorithm. Chapter 5 presents the importance-weighted stability Φ_{iw} , which is shown to be more suitable than current state-of-the-art stability measures in the context of stability optimization. Next, Chapter 6 addresses the issue of assessing stability in the presence of feature similarities. It also presents Φ_{msi} , which correctly deals with feature similarities without suffering from the major drawbacks of existing measures. The interactive visualization tool for domain analysis based on this last measure is introduced in Chapter 7. Finally, Chapter 8 concludes this thesis.

Bibliographic notes

Journal Publications

1. V. Hamer and P. Dupont. “An Importance Weighted Feature Selection Stability Measure”. In: *Journal of Machine Learning Research* 22.116 (2021), pp. 1–57.

Conference Publications

1. V. Hamer and P. Dupont. “Joint optimization of predictive performance and selection stability.” In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. 2020, pp. 381–386.
2. V. Hamer and P. Dupont. “Robust Selection Stability Estimation in Correlated Spaces”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021, pp. 446–461.

Workshop Papers

1. V. Hamer and P. Dupont. “Explicit Control of Feature Relevance and Selection Stability Through Pareto Optimality”. In: *Proceedings of the 3rd International Tutorial & Workshop on Interactive Adaptive Learning (IAL2019) in conjunction with ECML-PKDD 2019*. Vol. 1. 2019, pp. 64–79.

Miscellaneous Contributions

1. V. Hamer. *stabilitymaps: A domain analysis tool for feature selection*. R package. 2021. URL: <https://bitbucket.org/vhamer/stabilitymaps>.

Acknowledgments

I would like to express my deepest gratitude to my advisor Pierre Dupont. His careful guidance and support made this thesis possible. He helped me see the big picture when I got lost in inconsequential details. His numerous and insightful comments on my article drafts showed his major implication and greatly helped me to publish my work. I must admit this was somewhat frustrating at times but the end result was worth it without a doubt. He definitively helped me improve my writing skills as well as my critical and big-picture thinking.

I also thank my co-advisor Benjamin Haibe-Kains and thesis committee member Michel Verleysen for their precious comments and insights. In particular, the data Benjamin made available for me was a source of motivation for the development of the visualization tool described in Chapter 7. One of Michel's questions during a seminar also drove me towards the issue of stability estimation in correlated spaces which was the research I enjoyed the most.

My thanks also go to my colleagues and Machine Learning Group researchers Alexander Gerniers and Vincent Branders. Our regular team meetings and discussions were a source of inspiration. They also kindly accepted to proofread the present document.

I am also grateful to all the people in INGI for the great atmosphere in the department. The social events, lunchtime belote games and afternoon kicker breaks all contributed to my enjoyment of the time I spent at INGI.

I also thank the UCL infrastructure and the CECI in particular. Their computation clusters were essential in the successful conduct of this thesis.

Contents

Preamble	i
Acknowledgments	vii
Table of Contents	ix
1 Introduction	1
1.1 Machine Learning	1
1.2 Microarray data	2
1.3 Motivation for feature selection	4
1.4 The issue of selection instability	5
2 Background	9
2.1 Classification	9
2.1.1 Linear Discriminants	9
2.1.2 Logistic Regression	10
2.1.3 Regularizers	12
2.1.4 Support Vector Machine	13
2.1.4.1 Hard-margin SVM	13
2.1.4.2 Soft-margin SVM	13
2.1.5 Nearest Neighbors	15
2.1.6 Decision Trees and Random Forests	16
2.1.6.1 Decision Trees	16
2.1.6.2 Random Forests	18
2.1.7 Neural networks and deep learning	19
2.2 Feature Selection	23
2.2.1 Filters	24
2.2.1.1 Golub’s Ratio	25
2.2.1.2 T-test	25
2.2.1.3 Mutual Information Maximization	26
2.2.1.4 Minimum Redundancy Maximum Relevance	27
2.2.1.5 The Relief	28
2.2.2 Embedded Methods	30
2.2.2.1 Logistic LASSO Regression	30
2.2.2.2 Elastic-Net Logistic Regression	31

2.2.2.3	Group LASSO	31
2.2.2.4	Recursive Feature Elimination	32
2.2.2.5	Feature Selection with Random Forests . . .	33
2.2.2.6	Feature Selection with Neural Networks . .	34
2.3	Selection Stability	36
2.3.1	Similarity-based subset measures	39
2.3.2	Frequency-based subset measures	40
2.3.3	The Nogueira Index	41
2.3.4	Weight and ranking based measures	42
3	A Bi-Objective Evaluation Framework	45
3.1	The objective-space	45
3.2	The Hypervolume	47
3.3	Expected satisfaction	48
3.4	Evaluation Protocol	52
3.4.1	Statistical tools of stability	54
3.4.2	Statistical tools of accuracy	56
3.4.3	Statistic tools of the expected satisfaction	60
3.5	A concrete example	68
3.6	Discussion	69
4	The hybrid-RFE algorithm	73
4.1	The Algorithm	73
4.2	Evaluation on microarray data	74
4.3	Alternative bias strategies	79
4.4	Conclusion	81
5	An Importance Weighted Stability Measure	83
5.1	Feature Stability Maps	83
5.2	Motivation for a new measure	85
5.3	An Importance Weighted Stability Measure	87
5.3.1	Definition	87
5.3.2	Evaluating Feature Importance	88
5.3.3	Properties and Reductions	90
5.3.4	Time Complexity	92
5.3.5	Sampling Distribution And Statistical Tools	94
5.4	Importance Weighted Decomposition	97
5.5	The hybrid-RFE Extended	100
5.6	Case Study: Decision Making for Cancer Diagnosis	102
5.7	Stability Analysis of Standard Feature Selection Methods . . .	107
5.7.1	The LASSO	107
5.7.2	The Elastic-Net	109

5.7.3	Logistic RFE	109
5.7.4	Random Forests	110
5.7.5	The Relief	110
5.7.6	Deep Feature Selection	113
5.8	Variability of Stability	118
5.9	Conclusion	119
6	Stability in Correlated Spaces	121
6.1	Problem Statement	121
6.2	Current Stability Measures	123
6.2.1	Sechidis Index	123
6.2.2	POGR Index	124
6.2.3	Yu Index: bipartite matching	125
6.3	Limitations of current measures	125
6.3.1	Toy scenarios	125
6.3.2	Simulation Studies I	128
6.4	Stability as Maximal Shared Importance	132
6.4.1	Definition	132
6.4.2	Illustration	133
6.4.3	Properties and Reductions	134
6.4.4	Simulation Studies II	138
6.5	Stability of Standard Selection Methods	145
6.6	The issue with the Sechidis Index	150
6.7	Conclusion	152
7	Interactive Feature Stability Maps: an R package	153
7.1	Overview	153
7.2	Demonstration on Leukemia diagnosis	154
8	Conclusion and Perspectives	165
A	Additional proofs	171
A.1	Proofs of Properties	171
A.2	Proofs of Theorems	177
B	Decision Making for Cancer Diagnosis	181
C	Importance Weighted Stability of Standard Feature Selection Methods	189

D Correction for chance in correlated spaces **199**

D.1 Context 199

D.2 Simulated data: revisiting scenario 3 202

D.3 Stability on microarray data sets 203

Introduction

| 1

This introductory chapter first presents the concept of machine learning and its associated notations in Section 1.1. Section 1.2 then reviews a machine learning problem that is central to this thesis: predictive modeling based on microarray data. The dimensionality of such microarray data is usually very high, which motivates for *feature selection*. Feature selection in high-dimensional spaces comes with challenges that are not fully solved yet, as detailed in Section 1.3. In particular, selection instability strongly limits the adoption of machine learning in sensitive areas such as the biomedical field. This thesis aims to mitigate this issue by providing tools to correctly estimate and optimize selection stability, for the purpose of promoting the broader use of machine learning.

1.1 Machine Learning

Traditional computational algorithms are a sequence of explicitly programmed instructions specifically organized to tackle a given problem. For instance, sorting algorithms take an unordered list of numbers as input and compute the associated ordering. The sorting logic is explicitly encoded in these algorithms by the programmer.

Unlike these approaches, machine learning (ML) algorithms are trained on input data. They apply statistical analysis to understand its structure and often learn to make associated predictions. An ML algorithm trained on a large bank of annotated animal pictures automatically learns what visual features separate them. It can then identify these animals in new pictures without being explicitly programmed to do so. In that sense, the machine *learns* from the data. This is illustrated in Figure 1.1. Machine learning has been successfully applied in many applications, such as in medicine, computer vision, product recommendation, email filtering, and other applications for which it is difficult to build traditional computational algorithms that perform the needed tasks.

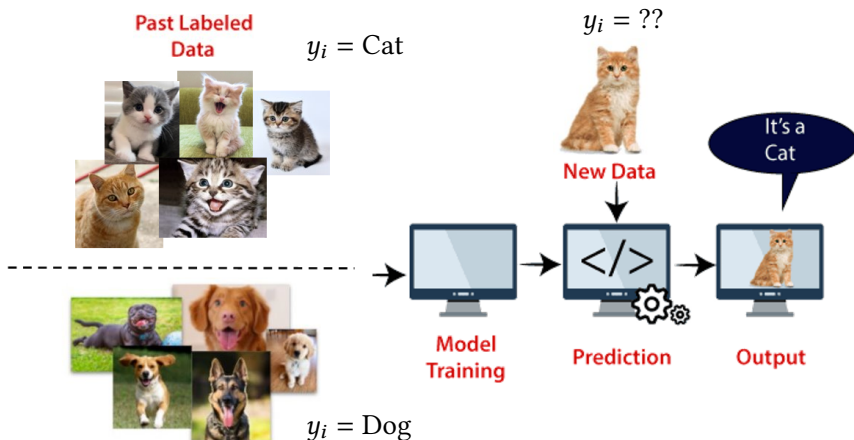


Figure 1.1: Illustration of a supervised binary classification task: image recognition of cats and dogs. Partially from [Jav].

Machine learning can be split into two fields: supervised and unsupervised learning. Supervised learning assumes that each data example (e.g., a picture) has an associated annotation (e.g., cat, dog, bird). More formally, supervised learning targets the estimation of a function $f : X \rightarrow Y$ that maps the input space X to some output space Y . This estimation is done by leveraging the information contained in a finite training set

$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$$

for which the output values are known. Supervised regression deals with a continuous output space Y . Supervised classification tackles the discrete case. In supervised classification, the target variable is called a label, and each element of Y is called a class. When Y is binary (e.g., $\{\text{cat}, \text{dog}\}$ in Figure 1.1), hence when there are only two classes, we talk about binary classification. Binary classification is the main focus of this thesis. Still, most of the methods detailed in this document are trivially extendable to handle multiple classes. In all cases, the input data X can be composed of continuous and discrete variables or even more complex data types, such as graphs.

In unsupervised learning, the data is unlabeled, and the algorithm must find similarities among the input data X . A prime example where unsupervised learning is valuable is anomaly detection whose target is to identify erroneous or abnormal values in a data set.

1.2 Microarray data

This section details a sensitive supervised classification task: the development of prognosis/diagnosis kits based on microarray gene expression profiles. Mi-

croarray data sets (among others) are used throughout this thesis to evaluate machine learning algorithms.

Microarray is a technology that allows the simultaneous measurement of the expression levels of thousands of genes. A DNA microarray is a glass slide on which microscopic DNA spots are attached at specific locations. These spots contain small amounts of a specific DNA sequence, called a probe [SMS03]. Without going into the specifics, when put in contact with the target DNA, the target sequences that are complementary to some probes tend to bind with them. If the target sequences are fluorescently labeled, they generate a signal that can be measured and used to estimate the relative abundance of nucleic acid sequences (*i.e.*, gene expressions) in the target. By simultaneously measuring the expression levels of thousands of genes, microarray technology has allowed researchers to tackle issues that were previously out-of-reach [Sch+95; DIB97].

Gene expression levels are likely to influence the produced amount of the corresponding proteins (*i.e.*, the proteins for which the DNA sequences code), which are critical ingredients of most biological processes in the human body.

Table 1.1 depicts a generic microarray input data under matrix form, along with a generic target variable. Each row represents a given data example $\mathbf{x}_i \in \mathbb{R}^d$ which consists of the gene expression levels of d probesets (*i.e.*, the columns) obtained via a single microarray experiment. The probeset expressions are found by aggregating the fluorescence readings of several (typically 10) probes. Each probeset represents a gene or a portion of genetic material that might play some role in the metabolism. The number of probesets d can be up to several tens of thousands if high-density arrays are used. In contrast, the number of rows n , the number of patients on which the study has been performed, rarely exceeds a few hundreds. The last column holds the labels, which have been manually annotated for each microarray experiment. For instance, if the microarray experiments have been performed on patients diagnosed with breast cancer, this label could indicate whether or not each patient experienced a cancer relapse. In this context, the goal of a machine learning algorithm is to learn a function $f : \mathbb{R}^d \rightarrow \{\text{relapse, no relapse}\}$ that is a good approximation of the true (and unknown) mapping function. One can then apply this knowledge to build a prognosis tool discriminating between breast-cancer patients with a low or high risk of cancer relapse.

	probeset 1	probeset 2	...	probeset d	label
patient 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,d}$	y_1
patient 2	$x_{2,1}$	$x_{2,2}$	...	$x_{2,d}$	y_2
...	...	...	...	...	...
patient n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,d}$	y_n

Table 1.1: Matrix representation of gene expression levels measured on n patients by microarrays with d probesets. The last column represents the labels which are annotated to each patient.

1.3 Motivation for feature selection

Microarray data are a prime example where the number of input features d is far larger than the number of data samples n . Such a scenario is often referred to as the *small- n -big- d* setting. This setting often motivates for *feature selection*. Feature selection can intuitively be defined as the identification of a small subset of relevant and informative input features. Feature selection brings several advantages, especially in high-dimensional input spaces [TAL14; SIL07; KPH07].

A. Feature selection reduces the cost of predictive kits The goal of machine learning is to learn predictive models. These models are then used on new data to make actual predictions. This requires the measurement of *every* input feature for each new data sample, which can be prohibitive. For instance, measuring the expression levels of all genes would make the use of diagnosis kits in clinics extremely costly. Drastically reducing the number of features that are actually used to make predictions is an efficient way to decrease run-time costs. Furthermore, it also speeds up the time needed to make predictions, which can be a bonus in time-sensitive tasks.

B. Feature selection alleviates the curse of dimensionality The curse of dimensionality designates the exponential growth of the number of samples needed to estimate an arbitrary function with respect to its number of input variables. The term was first introduced in the dynamic programming context to denote the volume increase of Euclidean space which comes with the addition of extra dimensions [Bel61]. In the machine learning context, the curse of dimensionality relates to the exponential decrease of the density of data points when the number of input features increases. For *small- n -big- d* problems, the combination of a high number of features and a low number of data samples is such that most of the input space is empty. The curse of dimensionality makes generalization to new data intrinsically difficult. Finding

predictive models that explain perfectly the training data becomes increasingly *easier* when space becomes *emptier*. For instance, a simple linear classifier perfectly classifying the training data is guaranteed to exist whenever $d+1 \geq n$ (provided that there is no collinear pair of points). In a *small-n-big-d* setting, $d \gg n$, and there exists a *huge* number of models that work perfectly on the training data. In this situation, knowing which models would also work well on new, unseen data becomes difficult. Generalization performance usually drops massively. Such a phenomenon is called *overfitting*. Feature selection can help limit such overfitting by drastically reducing the number of input features so that the points' density remains sufficient. The removal of noisy and irrelevant features can also increase predictive performance.

C. Feature selection facilitates domain analysis Domain analysis is sometimes performed on the predictive models learned via machine learning. In the biomedical domain, for instance, this can help get further insights into biological processes, how they work together, and how they influence the predictions. Feature selection helps in that regard, as it produces much smaller models that are easier to interpret and validate.

1.4 The issue of selection instability

Feature selection is broadly used in the scientific community and belongs to the toolbox of any respectable data scientist. Yet, feature selection still often suffers from a significant drawback: the lack of *selection stability*. Selection stability has been first defined in 2005 [KPH05]. It has brought more and more attention from the scientific community ever since, notably for microarray classification [Dav+06; Qiu+06; SK10; YHB11]. Selection instability arises when the selected features drastically change after minor data modifications (e.g., different sampling of patients in biomedical studies). This instability negatively impacts the interpretation of the selected features while also making domain experts skeptical about machine learning algorithms. The same biological markers are expected to be informative of the clinical outcome, independently of the group of patients considered. If slight changes in the sampling of the patients largely influence the selected features, they are unlikely to accurately represent the true underlying processes at play. Selection stability is important as it effectively quantifies how much a feature selection algorithm can be trusted. Domain experts often prefer more stable and trustworthy feature selection algorithms over unstable ones, even if the latter are slightly more accurate [KPH07; SAV08a]. This is particularly true in the biomedical field, where reproducibility is a key issue [Hai+13; Lee+13]. Selection stability is sometimes even considered to be equally important as predictive performance [Jur+08]. The lack of trust from domain experts due

to selection instability hinders the use of machine learning in the biomedical domain, among other fields, even though these fields could greatly benefit from machine learning.

While feature selection can sometimes reduce the overfitting caused by the curse of dimensionality, the latter is the prime reason behind selection instability in the *small-n-big-d* setting. In particular, stability heavily depends on the *sample-to-feature ratio* $\frac{n}{d}$ [Ale13]. This is confirmed later in our experiments. Stability can become extremely low when the *sample-to-feature ratio* approaches zero, which is often the case in microarray studies. Still, instability is not restricted to low *sample-to-feature ratios*. It remains an issue for a broad range of problems for which the interpretability of machine learning models is an important aspect.

Selection stability has brought an increasing amount of attention from the scientific community in recent years. These works can be split into two categories:

1. *Stability estimation* seeks to adequately quantify the degree of stability of feature selection algorithms.
2. *Stability optimization* aims to increase selection stability.

Both angles are necessary for domain experts to trust feature selection algorithms. The goals and contributions of this thesis incorporate these two aspects. In particular, Chapter 3 first presents a bi-objective evaluation framework to assess the aggregated quality of feature selection. Chapter 4 concentrates on the *optimization* of current state-of-the-art stability measures. It proposes the hybrid-RFE algorithm, which optimizes stability explicitly via a dedicated meta-parameter. This algorithm produces Pareto-optimal curves in the (accuracy, stability) objective space, from which domain experts can choose their favorite trade-off based on their personal preferences for predictive accuracy and selection stability.

The next two chapters mainly focus on *stability estimation*. It turns out that directly optimizing stability leads to situations where current stability measures are high even though the feature selection algorithm should *not* be considered trustworthy. This indicates that current measures are not adequate for stability optimization. Chapter 5 illustrates and solves this issue by introducing a novel and more appropriate stability measure in the context of stability optimization.

Aside from low *sample-to-feature ratios*, feature *redundancy*, also called feature *correlation*, is another main cause for selection instability [SN10a]. With that in mind, Chapter 6 introduces the specific challenges of stability estimation in the presence of high correlations between the input features. It illustrates the limitations of current stability measures and proposes a second

novel measure, shown to be more robust than existing alternatives. This work is again particularly relevant but not restricted to microarray data as they often exhibit strong feature redundancies.

Finally, Chapter 7 focuses on domain analysis. It provides an interactive visualization tool to facilitate domain analysis in the form of an R package.

This chapter provides the required background information on machine learning and feature selection for the non-initiated reader to follow the rest of this thesis. First, Section 1.1 introduces some well-known machine learning approaches. Then, feature selection, a sub-field of machine learning that aims to identify small subsets of informative features, is presented in Section 2.2 along with common selection methods. Finally, the notion of selection stability and current stability measures are discussed in Section 2.3.

2.1 Classification

This section reviews different methods and principles of supervised classification used throughout this thesis. We restrict ourselves to the analysis of binary classification. The presented techniques serve as bases for the feature selection algorithms introduced in Section 2.2.

2.1.1 Linear Discriminants

We assume the input space X to be the space of d -dimensional real-valued vectors, i.e., $X = \mathbb{R}^d$. Let $f : \mathbb{R}^d \rightarrow \{+1, -1\}$ be the decision function, the integers $+1$ and -1 representing the two classes of the binary classification problem.

The discriminative approach models the decision boundary via a function $g(\mathbf{x})$ and poses $f(\mathbf{x}) = \text{sign}(g(\mathbf{x}))$. Concretely, the first class is predicted for an input $\mathbf{x} \in \mathbb{R}^d$ if $g(\mathbf{x}) > 0$, and the second class is predicted if $g(\mathbf{x}) < 0$. The set of points for which $g(\mathbf{x}) = 0$ is called the decision boundary. The function g is often referred to as a machine learning *model* or *hypothesis*.

Linear models, also called linear discriminants, assume that function g to be a linear function of the input $\mathbf{x}$:

$$g(\mathbf{x}) = w_0 + \sum_{f=1}^d w_f \times x_f$$

The vector $\mathbf{w}$ contains the $d + 1$ parameters of the linear model: a bias term w_0 and d weights w_f , one for each dimension (or feature) of the input. These

parameters have to be learned in the training phase such that the function g separates both classes as well as possible.

To simplify the notations, the data vector $\mathbf{x}$ is often extended with a first dimension x_0 always set to 1 such that the function g can be written simply

$$g(\mathbf{x}) = \sum_{f=0}^d w_f \times x_f = \langle \mathbf{w}, \mathbf{x} \rangle.$$

For linear discriminants, the decision boundary $g(\mathbf{x}) = 0$ is a d -dimensional hyperplane. Figure 2.1 depicts two binary machine learning tasks in $\mathbb{R}^2$ with several possible separating hyperplanes represented by dashed lines (as the input space X is only two-dimensional). The cross and circles represent the data points of each class. Every point to the left of these hyperplanes is predicted to belong to the cross class while points to their right are predicted to be of the circle class. In Figure 2.1a, the data is said to be **linearly separable** as there exists hyperplanes perfectly separating the data. Still, an infinite amount of possible perfectly separating hyperplanes exist, and different machine learning algorithms might converge towards different solutions. In Figure 2.1b, the data is no longer linearly separable, as some points are misclassified.

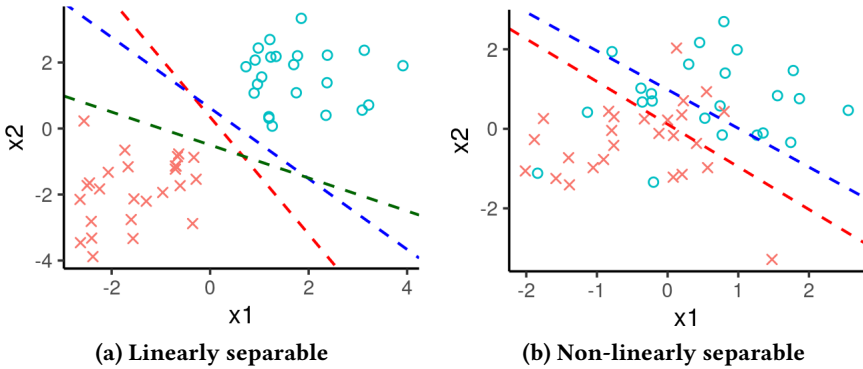


Figure 2.1: Examples of linearly separable (a) and non-linearly separable (b) machine learning tasks. The cross and circles represent the data points of each class in the two-dimensional input space X . The dashed lines are the decision boundaries (hyperplanes) $g(\mathbf{x}) = 0$ of several possible linear discriminants (blue: logistic regression, red: hard (a) and soft (b) margin SVMs, green: arbitrary).

2.1.2 Logistic Regression

Logistic regression learns linear models whose outputs are related to the probability of a data point belonging to either of the two classes (± 1).

Let the function σ such that $\sigma(g(\mathbf{x}))$ predicts the probability of the data point $\mathbf{x}$ to belong to the positive class: $\sigma(g(\mathbf{x})) = P(y = 1|\mathbf{x})$. Logistic regression first computes the *odds* of this event to happen. The odds of a binary event are defined as the ratio of the probability that this event will happen to the probability that the event will not happen. In our case,

$$\text{odds}(y = 1|\mathbf{x}) = \frac{P(y = 1|\mathbf{x})}{1 - P(y = 1|\mathbf{x})}. \quad (2.1)$$

This notion of odds is often used in gambling to compute the rewards in case of a correct prediction. One then imposes the logarithm of these odds to be equal to the function $g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$ computed by the linear discriminant :

$$\ln \frac{P(y = 1|\mathbf{x})}{1 - P(y = 1|\mathbf{x})} = g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle. \quad (2.2)$$

The left-hand side of Equation 2.2 is known as the *logit* function: $\text{logit}(p) = \ln \frac{p}{1-p}$ where p is a probability ($0 \leq p \leq 1$). The probability that the data point $\mathbf{x}$ belongs to the positive class is then found by isolating $P(y = 1|\mathbf{x})$ in Equation 2.2:

$$\sigma(g(\mathbf{x})) = P(y = 1|\mathbf{x}) = \frac{1}{e^{-\langle \mathbf{w}, \mathbf{x} \rangle} + 1}. \quad (2.3)$$

Figure 2.2 depicts the logistic function σ .

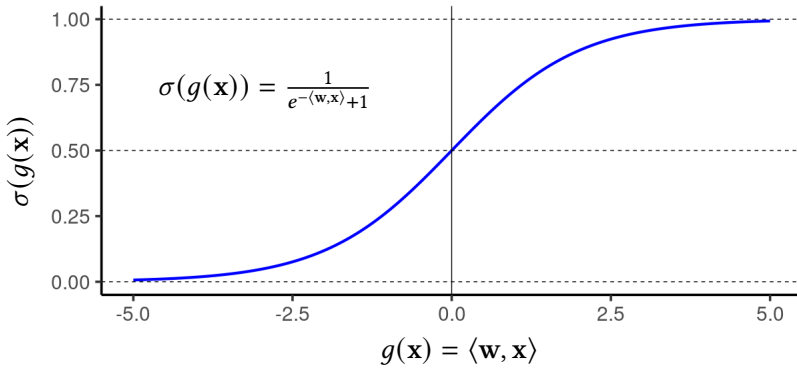


Figure 2.2: Logistic function. The data point $\mathbf{x}$ is predicted as being of the positive class if $\langle \mathbf{w}, \mathbf{x} \rangle > 0$, or, equivalently, if $\sigma(\langle \mathbf{w}, \mathbf{x} \rangle) > 0.5$.

The data point $\mathbf{x}$ is then predicted to belong to the positive class if the probability $\sigma(g(\mathbf{x}))$ is larger than 0.5. Using this new decision function or simply $f = \text{sign}(g(\mathbf{x}))$ leads to the same separating hyperplane. Indeed, one can easily verify that

$$(g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = 0) \Leftrightarrow \sigma(g(\mathbf{x})) = 0.5.$$

This situation corresponds to an equal probability of belonging to either of the two classes: $P(y = 1|\mathbf{x}) = P(y = -1|\mathbf{x}) = 0.5$.

The parameter vector $\mathbf{w}$ is usually estimated from the data using maximum likelihood estimation [Men02]. In essence, the chosen vector $\mathbf{w}$ is such that the observed data is most probable under the probabilistic interpretation developed here. The likelihood of a data point $\mathbf{x}_i$ is the probability, assuming our hypothesis $\mathbf{w}$ is correct, that the label of $\mathbf{x}_i$ is y_i . This probability is noted $P(y_i|\mathbf{x}_i; \mathbf{w})$. If $y_i = 1$, $P(y_i|\mathbf{x}_i; \mathbf{w}) = \frac{1}{e^{-\langle \mathbf{w}, \mathbf{x}_i \rangle} + 1}$. If $y_i = -1$, $P(y_i|\mathbf{x}_i; \mathbf{w}) = \frac{1}{e^{+\langle \mathbf{w}, \mathbf{x}_i \rangle} + 1}$. One can then write

$$P(y_i|\mathbf{x}_i; \mathbf{w}) = \frac{1}{e^{-y_i \langle \mathbf{w}, \mathbf{x}_i \rangle} + 1}. \quad (2.4)$$

Assuming the independence of data points, the likelihood of the whole data is the product of the individual probabilities $\prod_{i=1}^n P(y_i|\mathbf{x}_i; \mathbf{w})$. Maximizing this quantity or minimizing the logarithm of its inverse leads to the same solution and is usually easier to do. Hence the quantity to be minimized is

$$\ln\left(\prod_{i=1}^n P(y_i|\mathbf{x}_i; \mathbf{w})^{-1}\right) = \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x}_i \rangle}). \quad (2.5)$$

The parameter vector $\mathbf{w}$ of the linear discriminant is then:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ell_{\log}(y_i g(\mathbf{x}_i)) = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x}_i \rangle}) \quad (2.6)$$

with $\ell_{\log}$ known as the logistic loss. The solution to Equation 2.6 is found via an iterative procedure such as gradient descent.

The blue separating hyperplanes in Figure 2.1 are obtained with logistic regression.

2.1.3 Regularizers

It is common practice to add a *regularization* term to the loss function. This term penalizes the complexity of the hypothesis. The premise behind this practice is that a complex hypothesis might well separate the training data but is unlikely to generalize well to new, previously unseen data. The most common regularizers are the L_1 and L_2 norms:

$$L_1 = \|\mathbf{w}\|_1 = \sum_{f=1}^d |w_f|, \quad L_2 = \|\mathbf{w}\|_2 = \sqrt{\sum_{f=1}^d w_f^2}.$$

The model parameters are then found by minimizing a linear combination of the loss function and one or several regularizers. Logistic Ridge Regression uses the L_2 norm as regularizer which prevents the parameters $\mathbf{w}$ from taking

too high values. That generally improves the generalization capability of the linear discriminant. With Logistic Ridge Regression, the weight vector $\mathbf{w}$ is found according to Equation 2.7.

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x}_i \rangle}) + \lambda \|\mathbf{w}\|_2 \quad (2.7)$$

The blue hyperplane in Figure 2.1b has been obtained with this approach with $\lambda = 1$.

The LASSO uses the L_1 norm, which promotes sparsity (it forces some weights w_f to 0). This is a first example of feature selection algorithm, which *selects* the features f with non-zero weights w_f . This also often increases generalization by reducing the noise from irrelevant features. The LASSO and other feature selection techniques are covered in more detail in Section 2.2.

2.1.4 Support Vector Machine

Support Vector Machines [CV95] find the *maximum-margin hyperplane*. This maximum-margin hyperplane is the separating hyperplane for which the distance to the nearest point $\mathbf{x}_i$ of either class is maximized. An example is the red hyperplane in Figure 2.1a.

2.1.4.1 Hard-margin SVM

Assuming the data is linearly separable, consider the two parallel hyperplanes that fully separate the data such that their distance to each other is maximal. This distance is called the margin and the maximum-margin hyperplane is halfway from both of them. These three hyperplanes are depicted in Figure 2.3. As the equation of the hyperplane is unchanged if multiplied by an arbitrary positive constant, the scale is set by imposing $g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = \pm 1$ on the two parallel hyperplanes. The distance between these two planes, *i.e.*, the margin, is then $\frac{2}{\|\mathbf{w}\|_2}$. Maximizing the margin comes back to minimizing the L2 norm of $\mathbf{w}$. The data points that lie on one of the two separating hyperplanes, (the points for which $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle = 1$) are called support vectors. The other points all satisfy $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle > 1$ and do not influence the maximum-margin hyperplane. This maximum margin hyperplane is the solution of the optimization problem

$$\text{Minimize } \|\mathbf{w}\|_2 \text{ subject to } y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1, \text{ for } i = 1, \dots, n. \quad (2.8)$$

2.1.4.2 Soft-margin SVM

In many practical cases, the data is not linearly separable. Hard-margin SVMs assume linear separability and Equation 2.8 does not admit a solution if this

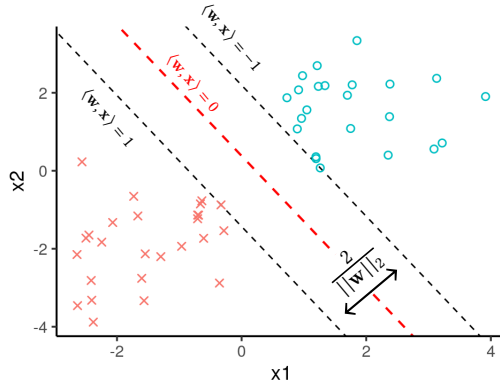


Figure 2.3: An instance of maximum-margin hyperplane (red). It lies exactly between the two most distant parallel hyperplanes that separate the data (black). The margin is the distance between these two hyperplanes.

assumption is violated. A workaround is to introduce a slack variable ξ_i for each data point $\mathbf{x}_i$, and relaxing the constraint of Equation 2.8 into

$$y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1 - \xi_i, \text{ for } i = 1, \dots, n. \quad (2.9)$$

The optimization problem then becomes

$$\min_{\mathbf{w}, \xi} \frac{1}{2} \|\mathbf{w}\|_2^2 + C \sum_{i=1}^n \xi_i, \quad (2.10)$$

subject to $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1 - \xi_i$, and $\xi_i \geq 0$ for $i = 1, \dots, n$.

The constant C balances between the original objective of margin maximization and the added objective of minimizing the margin violations. When C tends to ∞ , Equation 2.10 turns into Equation 2.8 and hard-margin SVMs, as no margin violation is permitted.

To minimize the objective of Equation 2.10, the slack variables ξ_i are either 0 if $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle \geq 1$, or are set to $1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle$ otherwise (the minimal value satisfying the constraints). One can reformulate them as $\xi_i = \max(0, 1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle)$. Optimization problem 2.10 is thus equivalent to

$$\min_{\mathbf{w}} \sum_{i=1}^n \max(0, 1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle) + \lambda \|\mathbf{w}\|_2^2 \quad (2.11)$$

with some constants manipulation. Equation 2.11 is analogous to Ridge Logistic Regression (Equation 2.7). SVMs and Ridge Logistic Regression differ only in the loss used: the first minimizes the logistic loss while the latter minimizes the *hinge loss* $\max(0, 1 - y \langle \mathbf{w}, \mathbf{x} \rangle)$. Figure 2.4 compares the logistic and hinge loss. The logistic loss never truly reaches zero, whereas the hinge loss is null for all points that do not violate the margins.

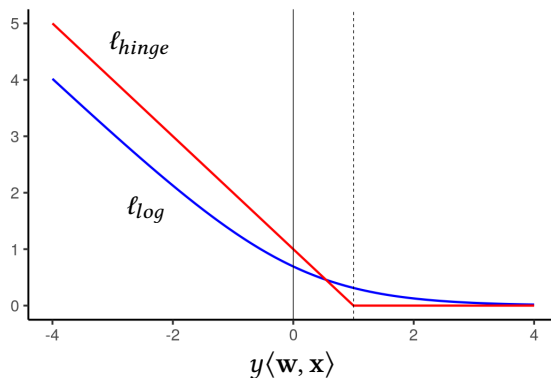


Figure 2.4: Logistic loss $\ell_{\log}$ and hinge loss ℓ_{hinge} .

2.1.5 Nearest Neighbors

Logistic Regression and linear Support Vector Machines produce linear decision models. They offer great interpretability but cannot model complex relationships between the input features and the target variable. A first example of a non-linear machine learning technique is the Nearest Neighbor algorithm (NN).

The algorithm is straightforward and requires no actual training. The predicted label of a previously unseen data point $\mathbf{x}$ is simply the label of its closest stored data point, *i.e.*, its closest neighbor:

$$\hat{y} = f(\mathbf{x}) = y_i : \mathbf{x}_i = \underset{i}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{x}_i\|_2. \quad (2.12)$$

A more robust approach is the k-Nearest Neighbor (kNN), where the predicted label is the label that occurs the most often among the K nearest neighbors of the test data point. Possible extensions exist where the vote is weighted by the distances between the test point and its K nearest neighbors.

Figure 2.5 depicts the decision boundaries that are obtained with the kNN algorithm with $K = 1$ (Figure 2.5a) and $K = 5$ (Figure 2.5b) on the same non-linearly separable data from Figure 2.1b. When $K = 1$, outliers can significantly alter the decision boundaries, such as in the ① and ② regions. It turns out that the best decisions in these regions are the opposite ones. Using $K = 5$ is more robust toward these outliers.

Nearest Neighbors techniques require no training time but might have a large prediction time. They are also highly susceptible to the curse of dimensionality. Indeed, potentially irrelevant features can largely influence the distance between points. Some form of dimensionality reduction, such as feature selection, is often required as a pre-processing step (or alternatively, some metric learning to learn a more adequate distance function than the

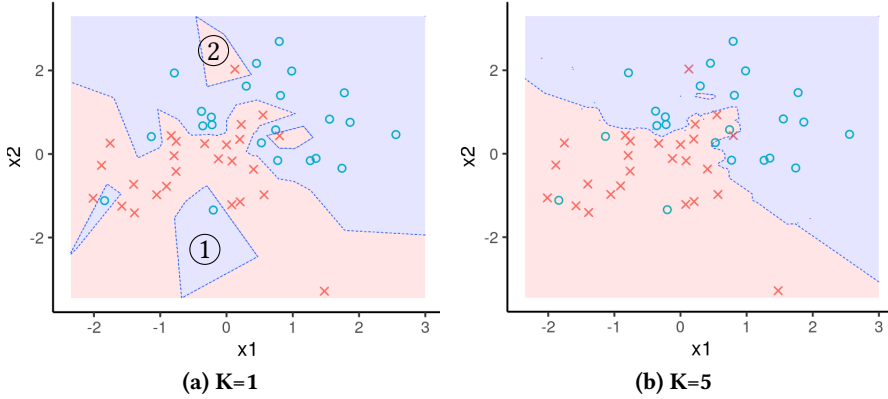


Figure 2.5: Decision boundaries that are obtained with the kNN algorithm with $K = 1$ (a) and $K = 5$ (b). Using more than a single neighbor increases robustness towards outliers.

classic Euclidian distance). The Relief algorithm is a feature selection approach based on the nearest-neighbor idea. It is presented in more detail in Section 2.2.1.5.

2.1.6 Decision Trees and Random Forests

This section reviews the popular machine learning approach known as Decision Trees. Random Forests combine several trees to build more robust and accurate predictors. Such Random Forests are the basis for another feature selection technique, studied in more detail in Section 2.2.2.5.

2.1.6.1 Decision Trees

Decision Trees are tree-shaped structures that assign a class label to each input example. Figure 2.6 depicts an example of a decision tree. Each node of the tree performs a test on some attribute value. Usually, binary tests are performed such that the input space is split in two at each node. Leaf nodes, *i.e.*, nodes with no children, contain the label predicted. For prediction, the data point goes through the tree, going down the branches corresponding to its attribute values until it reaches a leaf node, and the label of this node is predicted.

To learn decision trees from data, one cannot replicate the loss minimization approach used by many other machine learning algorithms, such as logistic regression or SVMs. Indeed, the number of possible trees is too high for the problem to be tractable. Instead, a greedy strategy that builds the tree step by step is used.

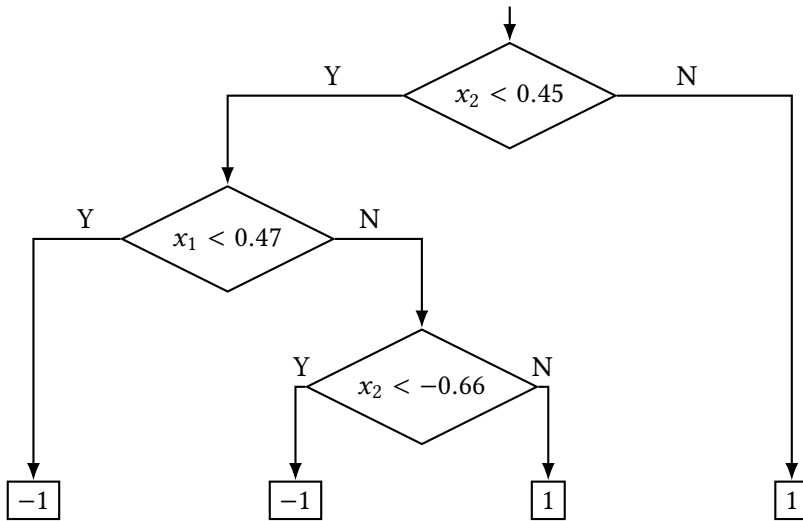


Figure 2.6: An example of a small decision tree learned on the running example. Its decision boundaries are depicted in Figure 2.7b

The tree construction starts with the root node corresponding to the whole feature space. The space is then recursively split in two as children nodes are added. To split a node o in two, one must decide 1) which feature f to evaluate and 2) what threshold value θ_o performs the split. Furthermore, some stopping criteria are often used to stop the tree from expanding too much.

A popular tree learning algorithm is the CART algorithm [Bre+84]. At each node, CART picks the feature f and threshold θ_o that maximizes the impurity decrease caused by the induced node splitting. The impurity of a node is measured by the Gini function:

$$\text{Gini}(S_o) = 1 - \sum_{y=1}^C p_{S_o}^2(y), \quad (2.13)$$

with C the number of classes and $p_{S_o}(y)$ the proportion of examples labeled y in the set of examples S_o associated to node o . For binary classification, $C = 2$, and Equation 2.13 turns into

$$\text{Gini}(S_o) = 1 - p_+^2 - p_-^2 = 1 - p_+^2 - (1 - p_+)^2 = 2p_+(1 - p_+). \quad (2.14)$$

A node is perfectly pure if all examples corresponding to it have the same class ($p_+ = 1$ or $p_+ = 0$), and impurity is maximal when there is no preference for a certain class ($p_+ = p_- = 0.5$). The drop in impurity is then the difference between the Gini value of node o and the sum of Gini values of both its

children, weighted by the relative fraction of examples associated to them:

$$\text{Drop}(S_o, f, \theta_o) = \text{Gini}(S_o) - \left(\frac{|S_{o_{\text{left}}}|}{|S_o|} \text{Gini}(S_{o_{\text{left}}}) + \frac{|S_{o_{\text{right}}}|}{|S_o|} \text{Gini}(S_{o_{\text{right}}}) \right), \quad (2.15)$$

with $S_{o_{\text{left}}} = \{\mathbf{x}_i \in S_o : x_{i,f} < \theta_o\}$ and $S_{o_{\text{right}}} = \{\mathbf{x}_i \in S_o : x_{i,f} \geq \theta_o\}$. The specific feature f and threshold θ_o are then chosen such as to maximize this drop of impurity, the node is split, the examples in S_o are each associated to one child and the procedure is applied recursively on both children.

If the here-above procedure is executed until each node contains only examples of the same class¹, *complete* trees are built. Learning complete trees usually leads to overfitting as the trees can become quite complex to classify every training example correctly. More robust decision trees can be learned by relying on some stopping criteria to stop branches from expanding too much. This is referred to as *pre-pruning*. In particular, in the original CART algorithm, a node becomes a leaf node if one of the two following conditions are met: the number of data examples at node o , $|S_o|$, is smaller than a certain threshold $\text{size}_{\min}$, or the impurity of the node, $\text{Gini}(S_o)$, is smaller than another threshold $\text{imp}_{\min}$. Slightly different versions of the original CART algorithm with different pruning mechanisms exist. One might want to impose a maximum depth to the trees or prevent node-splitting whenever the impurity drop $\text{Drop}(S_o, f, \theta_o)$ falls below yet another threshold $\text{drop}_{\min}$. Some post-pruning is sometimes considered as well to reduce the trees' complexity further.

Figure 2.7 depicts the decision boundaries of two decision trees on the running example. The first tree is a complete tree while the second is obtained with some pre-pruning ($\text{drop}_{\min} = 0.05$). Decision boundaries are parallel to the axes, and the instance space is split into rectangles.

2.1.6.2 Random Forests

Random Forests are an example of *ensemble*-based machine learning algorithms. Different decision trees are learned, and a majority vote between all of them decides the predictions. This generally increases the robustness of the predictions.

Each tree of the forest is learned on a different *bootstrap sample* of the input data. A bootstrap sample is a set of examples drawn randomly with replacement from the original data and of the same size n . Bagging learns a forest of trees learned on different bootstrap samples [Bre96]. For Random Forests, a second layer of randomness is added in the node-splitting process [Bre01a]. In CART, the feature inducing the maximal impurity drop is

¹This is achievable only if there is no data inconsistency.

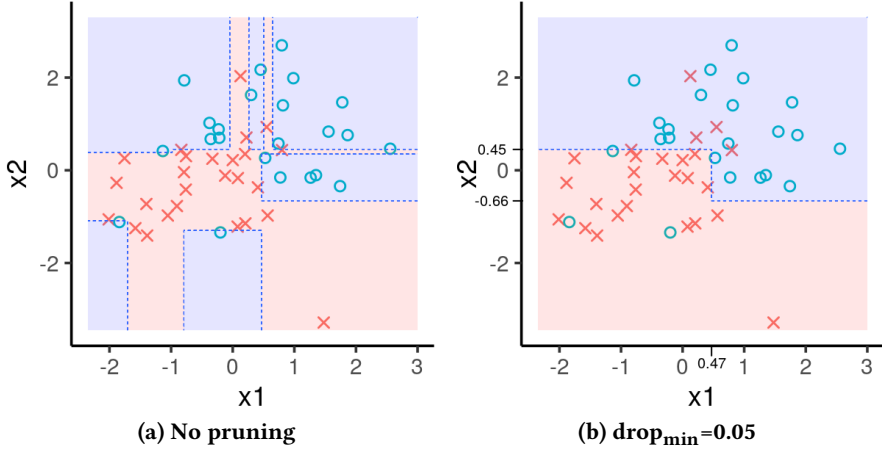


Figure 2.7: Decision boundaries that are obtained with the CART algorithm without any pruning (a) and with $\text{drop}_{\min}=0.05$ (b). The pruned tree is represented in Figure 2.6.

selected from all the input features to split the node. Random Forests first select mtry features randomly with equal probability and then choose the splitting feature from this set only. Trees in the forest are not pruned. Some additional randomization can be added to the determination of the splitting threshold. In that case, we talk about Extremely Randomized Trees [GEW06] instead of Random Forests.

Figure 2.8 depicts the decision boundaries of a Random Forest with 100 trees and $\text{mtry} = 1$ on the running example. Note that the interest of Random Forests is not at its maximum in a simple bi-dimensional space like in Figure 2.8. Indeed, since $\text{mtry} = 1$, the splitting feature is just chosen randomly, and only the decision threshold is optimized². Random Forests are the basis for another feature selection technique, studied in more detail in Section 2.2.2.5.

2.1.7 Neural networks and deep learning

Neural networks are another class of machine learning algorithms that have become increasingly popular in recent years. Neural networks are made of computational units called *neurons*. Such a neuron is represented in Figure 2.9. It first performs a simple linear combination of its inputs before applying a non-linear *activation function* to it. Each neuron has a single output. The simplest neural network, composed of a single neuron, can represent a

²Results with $\text{mtry} = 2$ are actually pretty similar as both features are equally relevant here.

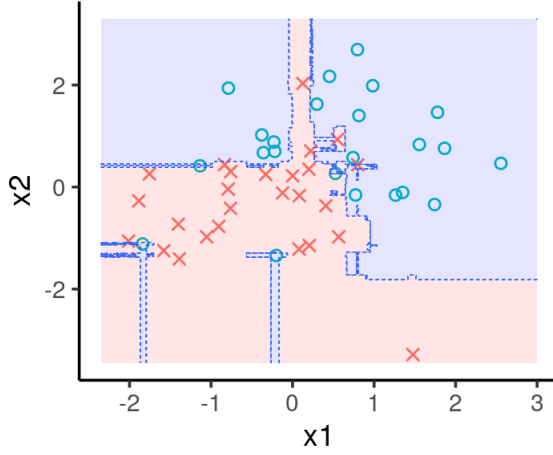


Figure 2.8: Random Forest boundaries on the running example. The forest is composed of 100 trees and `mtry` is set to 1.

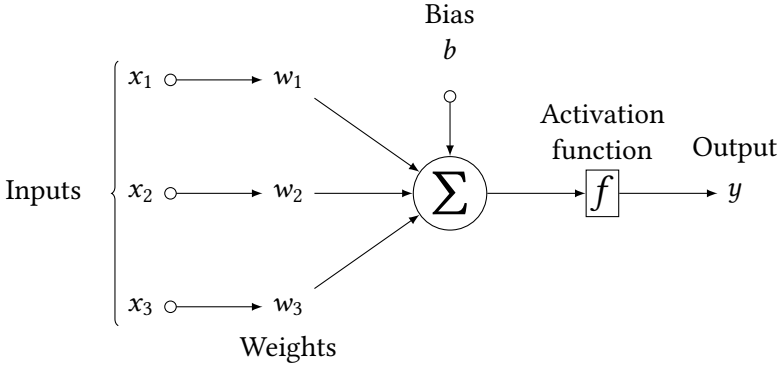


Figure 2.9: A single neuron. Neurons compute a linear combination of their inputs and then pass the result to a non-linear activation function.

linear classifier with the sign function as activation function. Its output is then

$$\hat{y} = f(\mathbf{x}) = \text{sign}(b + \sum_{f=1}^d w_f \times x_f).$$

The weights $\mathbf{w}$ and the bias b are parameters of the neuron and have to be learned from data.

More complex models can be designed by linking several neurons together and arranging them into *layers*, which form a neural *network*. Figure 2.10 represents such a network with two layers of neurons, often called *hidden layers*. The neurons of the first hidden layer each compute a linear combination of the d input features, apply a non-linear activation function

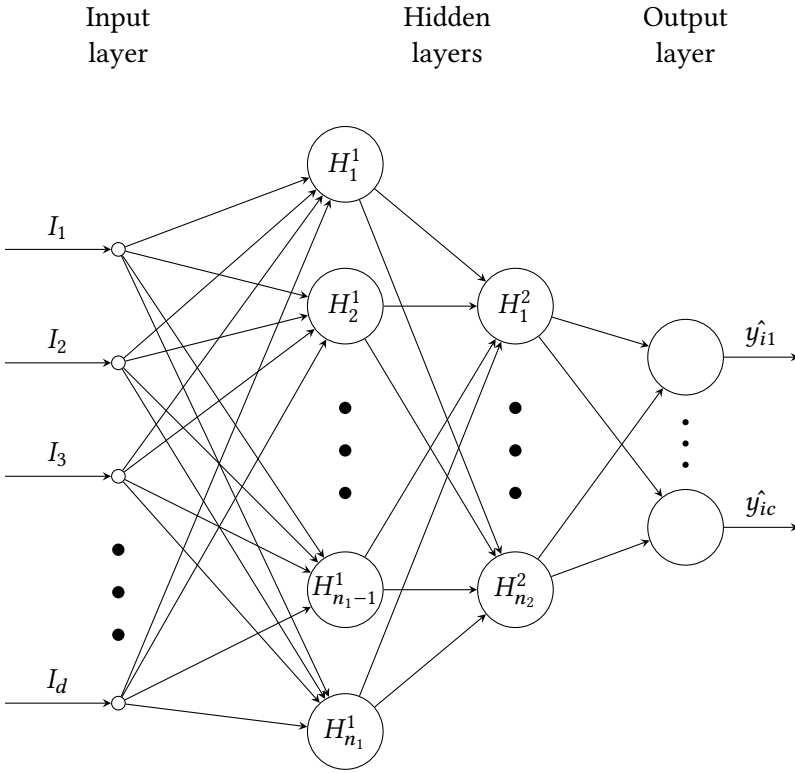


Figure 2.10: Feed-forward neural network with two hidden layers.

on the sum and pass the result to the next layer of neurons. The output layer is usually composed of the same number of neurons as the number of classes. The output of the output layer neurons is then an estimate of the probability of the input data being of the corresponding class. The term *deep learning* refers to the class of neural networks with many hidden layers, *i.e.*, deep neural networks. Many non-linear activation functions can be used; the important point is that they must be non-linear for the network to be able to model non-linear decision functions. Prominent examples include the sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$, the hyperbolic tangent $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ and the ReLU $\text{relu}(x) = \max(x, 0)$ activation functions.

Figure 2.10 represents the simplest type of deep neural networks, feedforward networks where the data flow goes only forward, towards the output layer. Furthermore, there is no imposed relation between the weight parameters of each neuron. More complex architectures exist, such as recurrent neural networks, convolutional networks, and generative adversarial networks. In particular, convolutional nets have proven to be particularly effective in image recognition tasks, which are outside the scope of this thesis.

To build an actual neural network, the weight parameters of each neuron are first initialized to small random values. Then, weight updates are iteratively computed via gradient descent in order to minimize a loss function $J(\Theta)$, where Θ represents all the parameters of the neural net, *i.e.*, a weight matrix W^k and a bias vector $\mathbf{b}^k$ in each layer k . A typical choice for the loss function $J(\Theta)$ is the cross-entropy loss

$$J(\Theta) = \sum_{i=1}^n \sum_{c=1}^C y_{ic} \log(\hat{y}_{ic}), \quad (2.16)$$

with y_{ic} the true label of example i ($= 1$ if the example i is of class c , and 0 otherwise); and with $\hat{y}_{ic}$ the output of the c th neuron of the output layer when example i is fed to the network. The cross-entropy loss becomes the binary cross-entropy loss

$$J(\Theta) = \sum_{i=1}^n y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \quad (2.17)$$

for binary prediction tasks. Regularizers, such as the L_1 and L_2 norms introduced in Section 2.1.3, can be applied on neurons as well for an expected reduced overfitting. Such regularizers create additional terms in the loss function $J(\Theta)$. Each update iteration is performed in two passes: a forward pass where the net's outputs $\hat{y}_{ic}$ are computed, and a backward pass where the gradients of J with respect to each network parameter are found. The gradients of the parameters of the last layer can be computed directly. Then, these gradients are *back-propagated* to the previous layer to compute this layer's gradients using the chain rule of calculus

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial h} \frac{\partial h}{\partial x}. \quad (2.18)$$

This is the *back-propagation* algorithm. After all gradients are computed, a weight update is performed in the direction of steepest descent, and the process is restarted until convergence or when a maximum number of iterations is reached. Using only a fraction of the training data to compute each weight update is common. The *batch size* refers to the number of data examples used for each update. Using a batch size smaller than n reduces the space requirements of the back-propagation algorithm. Furthermore, weight updates are applied faster, which usually reduces training time. However, the weight updates are less precise, which makes convergence towards the solution harder. When the batch size is set to one, one speaks about *stochastic gradient descent*. The number of *epochs* is the number of complete iterations, *i.e.*, the number of times each data example is propagated through the network to compute weight updates.

Figure 2.11 depicts the decision boundaries of a neural network without any hidden layer and a network with a single hidden layer of 64 units on the running example. For a complete overview of deep learning and its applications, we refer the reader to this book [GBC16].

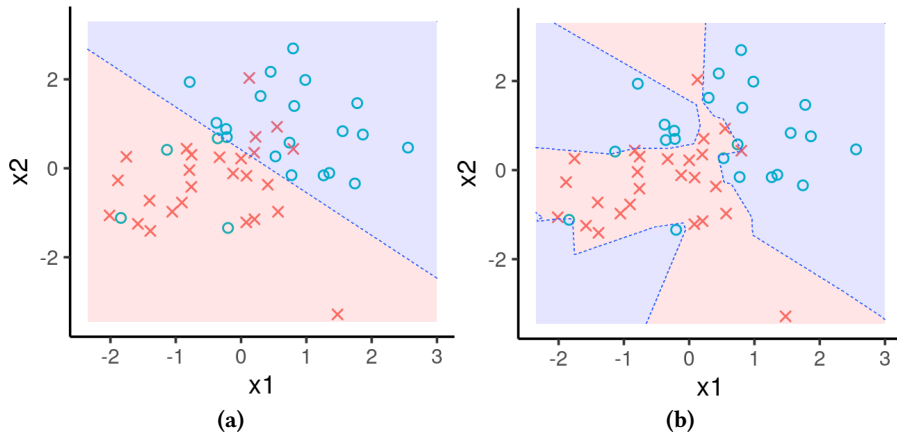


Figure 2.11: Neural network boundaries on the running example, without any hidden layer (a) and with a single hidden layer composed of 64 neurons with a ReLU activation function (b).

2.2 Feature Selection

This section introduces the notion of feature selection and reviews different feature selection methods, some of which are based on ML algorithms presented in Section 2.1.

Feature selection can be defined as the selection of a small subset of informative and relevant features, generally to be included in a predictive model. More formally, the output of a feature selection pipeline is a decision function $f : X_{\mathcal{F}} \rightarrow Y$ where $X_{\mathcal{F}} \subset X$ is the input space X restricted to the set of selected features $\mathcal{F} \subset \{1..d\}$. This is usually, but not always, done in two separate steps: 1) the selected feature set $\mathcal{F}$ is found by a dedicated feature selection algorithm, and 2) a standard ML technique is run on $X_{\mathcal{F}}$ and generates the predictive model. The term *feature selection* sometimes denotes only the first step of the here-above pipeline. However, for selection stability to be evaluated jointly with predictive accuracy, predictive models have to be learned.

Feature selection has become compulsory for a wide variety of applications due to the appearance of very high dimensional data sets, notably in the biomedical domain [SIL07]. Filtering noisy and irrelevant features can avoid overfitting the data and potentially improve predictive performance. Feature

selection also allows for the learning of fast and compact models that are easier to interpret. Such models can then be analyzed by domain experts and are easier to validate. Getting more interpretable models is also a key concern nowadays, and even considered by many as a requirement, when deployed in the medical domain.

Feature selection has already been studied in depth [TAL14; SIL07; KPH07]. Yet, current methods are still somewhat unsatisfactory mainly because of the typical instability they exhibit. Instability here refers to the fact that the selected features may drastically change after marginal modifications of the data or, more generally, after some fine-tuning of the data production or data analysis pipeline. This stability issue is central to this thesis and is further developed in Section 2.3.

Feature selection techniques are generally split into three categories: filters, wrappers, and embedded methods [LM98; Guy+08]. *Filters* evaluate the relevance of features independently of the final model, most commonly a classifier, and remove low-ranked features. Simple filters (e.g., t-test or ANOVA) are univariate, which is computationally efficient and tends to produce a relatively stable selection, but they plainly ignore the possible dependencies between various features. Information-theoretic methods, such as mRMR [DP05] and many others, are based on mutual information between features or with the response. However, a robust estimation of these quantities in high dimensional spaces remains difficult. *Wrappers* look for the feature subset that yields the best predictive performance on a validation set. They are classifier dependent and very often multivariate. However, they can be very computationally intensive, and an optimal feature subset can rarely be found. *Embedded methods* select features by determining which features are more important in the decisions of a predictive model. Prominent examples include SVM-RFE [Guy+02] and logistic regression with a LASSO [Tib96] or ELASTIC NET penalty [ZH05]. These methods tend to be more computationally demanding than filters, but they integrate the feature selection and the estimation of a predictive model into a single procedure. Nonetheless, they also tend to produce much less stable models. Recently, deep neural networks have started to be used as feature selectors as well [LCW16; RMM15]. This thesis focuses on filters and embedded methods, as wrappers are generally too computationally intensive when the feature space contains tens of thousands of features, such as for microarray data. Section 2.2.1 and 2.2.2 respectively review some popular filters and embedded methods that are used later in our experiments.

2.2.1 Filters

Filter feature selection techniques are independent of any machine learning algorithm. They usually compute a complete ranking of the input features,

from which the top- k features are selected and used in a second stage to construct a predictive model.

2.2.1.1 Golub's Ratio

The Golub's ratio measures the *signal to noise* ratio and has been applied by Golub et. al. on gene expression data [Gol+99]. It can be computed as:

$$G_f = \frac{|\mu_+^f - \mu_-^f|}{\sigma_+^f + \sigma_-^f}, \quad (2.19)$$

with μ_+^f and σ_+^f the average value and standard deviation of feature f among the examples of the positive class (among the negative class for μ_-^f and σ_-^f). The greater the Golub's ratio of feature f , the more relevant the feature. The *signal* corresponds to the numerator of Equation 2.19. The more the means of the two classes are separated, the better the signal. The *noise* refers to the denominator, which measures the standard deviation of the two classes. The bigger these deviations are, the less the two classes are separable, even if their means are far away.

2.2.1.2 T-test

The Welch's T-test statistic is similar to the Golub's ratio but has the advantage of presenting a statistical interpretation [Wel47]. It is used to assess if the observed difference between the means of the two classes is statistically significant. The null hypothesis, noted H_0 , is that both means are equal. Under this hypothesis,

$$T_f = \frac{|\mu_+^f - \mu_-^f|}{\sqrt{\frac{(\sigma_+^f)^2}{N_+} + \frac{(\sigma_-^f)^2}{N_-}}} \quad (2.20)$$

follows a Student's t-distribution. By measuring T_f on our data and looking into the t-distribution table, one can find the p-value of the statistical test. This p-value is the probability that the statistic measured on normally distributed data (with the same means and standard deviations) is greater than or equal to the one observed. Usually, the null hypothesis H_0 is rejected whenever this p-value is below 5%. In the examples of Figure 2.12, the p-value corresponding to $T_{x_1} = 10.14$ is $2.78 \cdot 10^{-9}$ and the one corresponding to $T_{x_2} = 5.86$ is $9.53 \cdot 10^{-7}$. Figure 2.12 illustrates the computation of the Golub's ratio and the Welch's T-test on a bi-dimensional example.

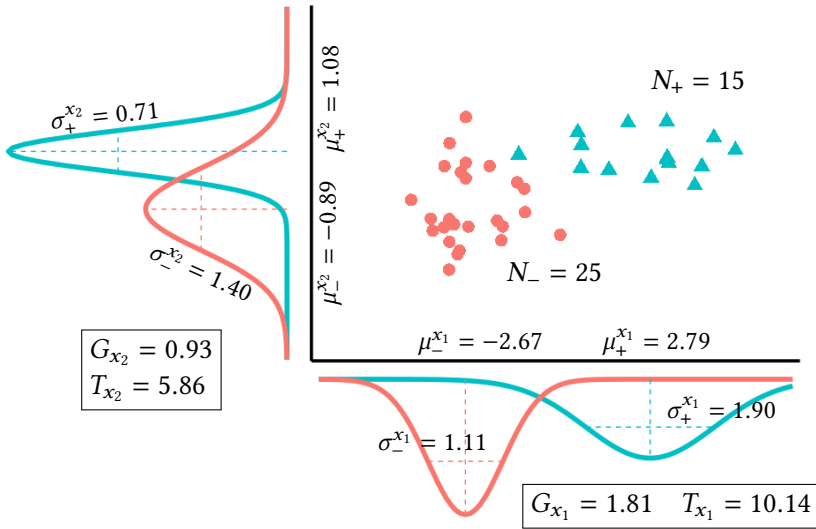


Figure 2.12: Example of computation of the Golub's ratio and the T-test on a bi-dimensional data set. The feature x_1 is more informative than x_2 according to both indices.

2.2.1.3 Mutual Information Maximization

In information theory, the mutual information of two random variable is a measure of their dependence. It measures the reduction in uncertainty for one variable given a known value of the other variable. It is defined as

$$I(X; Y) = H(X) - H(X|Y), \quad (2.21)$$

where $H(X)$ is the entropy of the random variable X

$$H(X) = - \sum_i p(v_i) \ln(p(v_i)), \quad (2.22)$$

where the v_i 's are the discrete values that X can take. The entropy $H(X)$ measures the information gain that is obtained by measuring the random variable X . $H(X|Y)$ is the conditional entropy of X and Y , i.e., the expected information gain obtained by measuring X if Y is already known. The conditional entropy $H(X|Y)$ can be shown to be equal to the difference between the entropy of the joint distribution of X and Y and the entropy of Y alone: $H(X|Y) = H(X, Y) - H(Y)$. The mutual information can then also be written as

$$I(X; Y) = H(X) + H(Y) - H(X, Y). \quad (2.23)$$

Figure 2.13 illustrates the relations between the different entropies. Continuous features can be discretized before computing their mutual information.

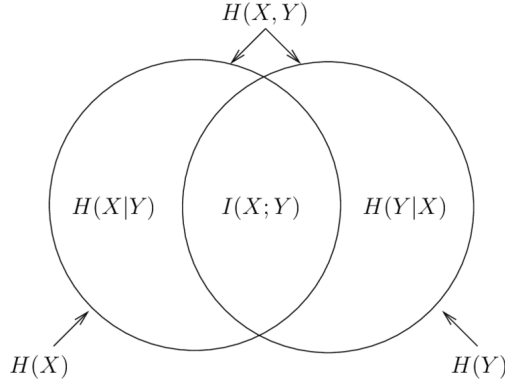


Figure 2.13: Illustration of the entropy H of different quantities and their relations.

For feature selection, features are ranked based on their mutual information with the class label. In other words, features are ranked based on how much their measurement reduces the uncertainty of the class label. If measuring feature f is enough to predict the class label with perfect accuracy, then the two circles in Figure 2.13 coincide, and mutual information is maximal: $I(X_f, Y) = H(Y)$. Conversely, if measuring f does not reduce the uncertainty of the class label at all, *i.e.*, when X_f and Y are *independent*, the two circles do not intersect, and the mutual information is zero. Mutual Information Maximization (MIM) ranks features based on their mutual information with the class label and selects the top- k features [Bro+12]. Feature selection based on mutual information is a powerful tool as it does not rely on any underlying assumption (such as normality of the distributions). Yet, its accurate estimation remains difficult when the number of samples is small.

2.2.1.4 Minimum Redundancy Maximum Relevance

The three filter techniques studied so far (Golub’s ratio, the T-test and Mutual Information Maximization) are *univariate* filters. They compute a score for each feature independently of the others and derive a full ranking from which the top- k features can be selected and used to build predictive models. The top features are all expected to be *relevant*, but they might be *redundant* with each other. The k best features are not necessarily the best k features.

The Minimum Redundancy Maximum Relevance (mRMR) algorithm seeks to find a feature subset that is *maximally relevant* but also *minimally redundant* [DP05; PLD05]. Finding the best feature subset according to this idea is intractable in high-dimensional spaces as it requires the evaluation of $\frac{d!}{k!(d-k)!}$ different feature sets. Instead, mRMR employs a greedy strategy. The

algorithm starts with an empty feature set and iteratively adds features to it until a pre-defined number k of selected features is reached. At each iteration i , the following score is computed for each eligible feature:

$$mRMR_i^f = \frac{\text{relevance}(f, Y)}{\text{redundancy}(f, \mathcal{F}_{i-1})}, \quad (2.24)$$

and the feature with the highest score is added to the set of currently selected features $\mathcal{F}_{i-1}$. This methodology is multivariate as the score of a feature f , $mRMR_i^f$, depends on some other features, here $\mathcal{F}_{i-1}$. Different relevance and redundancy measures can be used. One classic choice to quantify the relevance in the case of a categorical target variable is the F-statistic, which degenerates into the squared T-test T_f^2 for binary classification. To measure the redundancy between a candidate feature f and the set of already selected features $\mathcal{F}_{i-1}$, the averaged Pearson's correlation is a popular choice [DP05].

2.2.1.5 The Relief

The Relief algorithm produces a complete ranking of the features. Feature scores are computed based on the feature value differences between nearest-neighbor pairs [KR+92]. Whenever a feature value difference is identified between two nearest neighbors of the same class, a hit is recorded, and the feature score decreases. If such a difference is observed between nearest neighbors of different classes, a miss is recorded, and the feature score increases. The Relief algorithm is outlined in Algorithm 1. The feature scores are first all initialized to zeros. Then, an example x is selected at random m times (without replacement), with m usually set to the total number of training examples. For each sampled example x , the closest examples to x of the same and of different classes are found. They are respectively called the nearest hit and nearest miss. The feature scores are then updated according to line 7, with

$$\text{diff}(f, x_1, x_2) = \frac{|x_1^f - x_2^f|}{\max(f) - \min(f)}. \quad (2.25)$$

This function is also used to compute the distances between examples, which is required to find the nearest neighbors. The distance between two examples is simply the sum of the distances between their feature values (Manhattan distance). The denominator of Equation 2.25 ensures that feature distances are in the interval $[0, 1]$ such that the obtained feature scores are also bounded in the interval $[-1, 1]$.

A popular extension is the ReliefF algorithm [Kon94] which finds the K nearest hits and misses and averages their contribution to the score updates.

Algorithm 1 Relief

```

1: procedure RANKFEATURES( $m$ )
2:    $\text{score}_f \leftarrow 0, \forall f$ 
3:   repeat  $m$  times
4:      $x \leftarrow$  sample an example at random
5:      $\text{nearHit} \leftarrow$  closest same-class example from  $x$ 
6:      $\text{nearMiss} \leftarrow$  closest different-class example from  $x$ 
7:      $\text{score}_f \leftarrow \text{score}_f - \frac{1}{m} \text{diff}(f, x, \text{nearHit}) + \frac{1}{m} \text{diff}(f, x, \text{nearMiss}), \forall f$ 
8:   return score

```

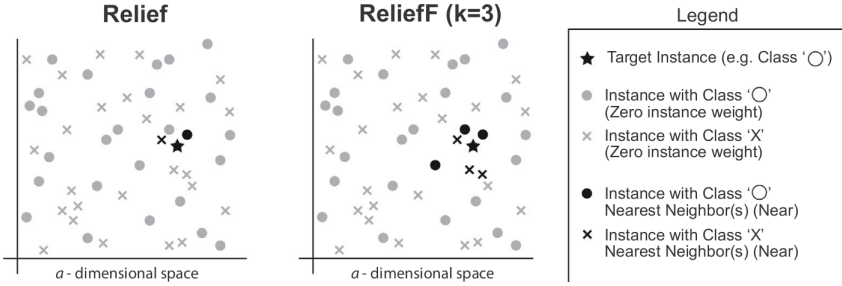


Figure 2.14: Illustration of the Relief (left) and ReliefF with $K = 3$ (right) algorithm. The star corresponds to the randomly drawn data example x in Algorithm 1. Assuming x is of the circle class, the three circles in black are the nearest misses while the three black crosses are the nearest hits. Image from [Urb+18].

The score updates (line 7 of Algorithm 1) becomes

$$\text{score}_f \leftarrow \text{score}_f - \frac{1}{Km} \sum_{j=1}^K \text{diff}(f, x, \text{nearHit}_j) + \frac{1}{Km} \sum_{j=1}^K \text{diff}(f, x, \text{nearMiss}_j) \quad (2.26)$$

with nearHit_j and nearMiss_j the j th nearest hit and miss. The motivation behind this approach is that redundant and noisy features may strongly affect the selection of the nearest neighbors, especially in high-dimensional spaces. The selection of the hits and misses is critical for the algorithm to identify relevant features, and averaging over K neighbors alleviates this issue. Yet, this problem of noisy neighbor identification is still present and worsens when the number of noisy features increases [RK03], *i.e.*, neighbor selection becomes increasingly random. Iterative Relief tackle this issue further by running the standard algorithm several times. Each time, the feature scores from the previous iteration are used in the distance computations, such that the impact of noisy features in the identification of the nearest neighbors decrease from iteration to iteration [DKB03].

Figure 2.14 illustrates the nearest hits and misses identification for the original Relief and the Relieff algorithm. Many other Relief variants exist. Some use different distance functions than in Equation 2.25; others give different weights to the K nearest hits and misses in the weight updates. We refer the reader to this extensive review [Urb+18].

2.2.2 Embedded Methods

Filters are independent of any machine learning algorithm used to build predictive models on the selected features. They are computationally efficient but may not always return a suitable feature subset for a specific predictive algorithm. *Embedded methods* select the features that are the more important in the decisions of a particular predictive model. This section reviews popular embedded techniques.

2.2.2.1 Logistic LASSO Regression

The LASSO algorithm has been originally proposed in the context of classical regression. It combines the square loss for regression and the L_1 norm regularizer [Tib96]. The parameter vector $\mathbf{w}$ of the linear model is found through Equation 2.27:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n (y_i - \langle \mathbf{w}, \mathbf{x} \rangle)^2 + \lambda \|\mathbf{w}\|_1. \quad (2.27)$$

This LASSO penalty can also be applied to generalized linear models [Lok99; Rot04], for instance to logistic regression [GLM07; SK03]. Logistic LASSO Regression combines the logistic loss (introduced in Section 2.1.2) with the L_1 norm regularizer $\|\mathbf{w}\|_1 (= \sum_{f=1}^d |w_f|)$. It finds the linear model $\mathbf{w}$ according to Equation 2.28:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x} \rangle}) + \lambda \|\mathbf{w}\|_1. \quad (2.28)$$

The L_1 norm is a sparsity-inducing regularizer: it forces some of the feature weights w_f to zero. The LASSO thus estimates a predictive model and performs feature selection (by discarding features with a zero weight) in a single step. The larger the regularization parameter λ , the fewer features are selected.³

³Sparsity is induced because the derivative $|\frac{\partial(\lambda \|\mathbf{w}\|_1)}{\partial w_f}|$ is constant (and equal to λ). In particular, there is no diminishing return when w_f approaches zero. Rising λ increases the incentive to shrink some weights, possibly to zero, reducing the number of selected features.

The LASSO comes with a set of limitations [ZH05]. First, it is well-known for being unstable. In particular, it tends to select one feature randomly when several of them are highly correlated to each other. This phenomenon is studied in more detail in Sections 6.3.2 and 6.4.4. On the other hand, Logistic Regression with a Ridge penalty (the L_2 norm $\|\mathbf{w}\|_2$) shrinks the weights of correlated features together and keeps all of them, but it does not promote sparsity. Furthermore, whenever the number of features d exceeds the number of data point n , the LASSO can select at most n features before it saturates [Efr+04]. Both of these properties can be detrimental in microarray studies. The correlations between genes sharing the same biological pathway can be high [SDC03], making the LASSO particularly unstable. Furthermore, the small number of patients n usually included in those studies strongly limits the number of selected features.

2.2.2.2 Elastic-Net Logistic Regression

The ELASTIC NET penalty is a direct generalization to the LASSO penalty that does not suffer from the limitations highlighted above. It minimizes a linear combination of the LASSO and Ridge penalties, keeping the best of both worlds. It promotes sparsity, tends to select correlated features together, and has no limitations in terms of number of selected features. The Elastic-Net Logistic Regression finds $\mathbf{w}$ following

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x} \rangle}) + \lambda_1 (\lambda_2 \|\mathbf{w}\|_1 + \frac{(1 - \lambda_2)}{2} \|\mathbf{w}\|_2). \quad (2.29)$$

The λ_1 meta-parameter dictates the intensity of the regularization, while λ_2 balances between the LASSO ($\|\mathbf{w}\|_1$) and Ridge ($\|\mathbf{w}\|_2$) penalties. For $\lambda_2 = 1$, the Elastic-Net becomes equivalent to the pure LASSO. Whenever $\lambda_2 = 0$, it becomes the standard Ridge regression.

2.2.2.3 Group LASSO

The Group LASSO takes advantage of prior knowledge about the structure of the feature space [YL06]. More specifically, it relies on user-defined feature groups. This is particularly relevant in genomic analysis as gene groups are known to work together to form pathways. The Group LASSO for logistic regression [MVB08] minimizes

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^n \ln(1 + e^{-y_i \langle \mathbf{w}, \mathbf{x} \rangle}) + \lambda \|\mathbf{w}\|_{1,2}, \quad (2.30)$$

with $\|\mathbf{w}\|_{1,2}$ the $L_{1,2}$ norm, some form of intermediate between the L_1 and L_2 norms:

$$\|\mathbf{w}\|_{1,2} = \sum_{i=1..G} \sqrt{\sum_{f \in \mathcal{G}_i} w_f^2}. \quad (2.31)$$

A Ridge penalty is applied to the features within each of the G user-defined feature groups. Furthermore, a LASSO penalty is applied on top, at the level of the groups. Consequently, this $L_{1,2}$ penalty promotes sparsity among the groups, but the features inside each group are all selected together.

The Group LASSO requires the feature groups to be disjoint. Extensions that allow the feature groups to intersect have been proposed [JOV09; ZRY09]. The Tree-LASSO is another generalization of the Group LASSO where the feature space is represented by a hierarchical tree of feature groups [KX10]. In general, ensuring that features inside correlated groups are selected as a whole increases selection stability. The DRAGS algorithm is another approach that automatically discovers dense feature groups and treats them as single selectable units for the purpose of stability improvement [YDL08].

The optimal weights vector of equations 2.7, 2.28, 2.29 and 2.30 can be found efficiently by coordinate descent algorithms [FHT10].

2.2.2.4 Recursive Feature Elimination

The LASSO, Elastic-Net, and Group LASSO perform feature selection by discarding features with a zero weight in the learned predictive model. They incorporate feature selection and model estimation in a single joint step thanks to the presence of the L_1 norm in their respective loss functions. Feature selection can also be performed a posteriori by estimating the importance of each feature in the predictive model and keeping the most important ones. The Recursive Feature Elimination algorithm is an iterative procedure that estimates predictive models and drops the feature with the lowest importance at each iteration until a user-defined number k of features is reached.

The Recursive Feature Elimination methodology has been originally proposed with SVMs as the model estimation algorithm [Guy+02]. The importance of a feature in those SVMs is defined as its contribution to the margin of the SVM, or rather, as the margin decrease the absence of the feature would cause. Importance is evaluated using Equation (2.32)

$$\mathcal{I}_{\text{svm}}(f, \boldsymbol{\alpha}) \propto |W^2(\boldsymbol{\alpha}) - W_{(-f)}^2(\boldsymbol{\alpha})|, \quad W_{-f}^2(\boldsymbol{\alpha}) = \sum_{k,l} \alpha_k \alpha_l y_k y_l \langle \mathbf{x}_k^{-f} \mathbf{x}_l^{-f} \rangle, \quad (2.32)$$

where $\mathbf{x}_k^{-f}$ denotes the training point k without the feature f , y_k is the label of training point k (± 1) and the α_k 's are the solutions to the svm dual prob-

lem [Guy+02]. The authors show that in the specific case of linear SVMs, ranking features based on Equation 2.32 is equivalent to ranking features on the absolute value of their weight $|w_f|$.

This methodology is applicable to many model estimators, and in particular to the Ridge Logistic Regression. As Ridge Regression produces linear models, one can simply use $|w_f|$ as an importance measure. The resulting Logistic-RFE algorithm is outlined in Algorithm 2. When the number d of input features is large, it is common practice to drop a fixed fraction of the bottom ranked features at each iteration, instead of a single one. This reduces the running time of the algorithm.

Algorithm 2 Logistic RFE

```

1: procedure SELECTFEATURES( $\lambda, \lambda_f$ )
2:    $\mathcal{F} \leftarrow$  the set of all features
3:   while  $|\mathcal{F}| > k$  do
4:      $\mathbf{w}^* \leftarrow \operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^n \log(1 + \exp^{-y_i(\mathbf{w}\mathbf{x}_i)}) + \lambda \|\mathbf{w}\|_2$ 
5:      $\mathbf{r}^* \leftarrow$  rank features  $f \in \mathcal{F}$  on  $|w_f^*|$  in descending order
6:      $\mathcal{F} \leftarrow \mathcal{F} - \{f : r_f^* = |\mathcal{F}|\}$ 
7:    $\mathbf{w}^* \leftarrow \operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^n \log(1 + \exp^{-y_i(\mathbf{w}\mathbf{x}_i)}) + \lambda_f \|\mathbf{w}\|_2$ 
8:   return ( $\mathcal{F}, \mathbf{w}^*$ )
  
```

2.2.2.5 Feature Selection with Random Forests

Random forests can be used for feature selection as well. In a Random Forest, each decision tree is built on a different bootstrap sample of the original data. A bootstrap sample is a set of examples that have been drawn randomly with replacement from the original data, and that has the same size. A forest of T trees is built on T bootstrap samples. For a particular tree t , the out-of-bag examples OOB_t are the data examples that are not part of the bootstrap on which the tree t is built. Feature importance is computed by randomly permuting the features in the out-of-bag samples of each of the T trees and by estimating the predictive accuracy decrease these permutations imply [Bre01b; PVD+13]. The features whose removal causes the largest accuracy decreases are selected, and a new random forest is learned with those features only.

More formally, the importance of a feature f in a Random Forest is defined as

$$I_{\text{rf}}(f) = \frac{1}{T} \sum_{t=1}^T \frac{1}{|\text{OOB}_t|} \sum_{\mathbf{x}_i \in \text{OOB}_t} (\hat{y}_i^{\text{f.perm}} \neq y_i) - (\hat{y}_i \neq y_i), \quad (2.33)$$

with y_i the true label of example $\mathbf{x}_i$, $\hat{y}_i$ the label predicted by the tree t without any permutation, and $\hat{y}_i^{\text{f.perm}}$ the label predicted by the tree t after the values

of feature f have been randomly permuted among the out-of-bag examples of t . This feature importance evaluation methodology is naturally stochastic. The top- k features according to this criterion are then selected.

Other criteria, such as the mean decrease of Gini index after feature permutations, can be used. [PVD+13] defines a statistical importance measure similar to $\mathcal{I}_{\text{rf}}$. The measure computes the p-values of a statistical test, which is a natural threshold for deciding which features to keep. This prevents the requirement of setting a pre-defined number k of selected features. An approach using the Recursive Feature Elimination methodology can also be applied to Random Forests, where features are ranked according to Equation 2.33 and dropped iteratively [DD06], much like in Algorithm 2.

2.2.2.6 Feature Selection with Neural Networks

Recently, deep neural networks have started to be used as feature selectors. Li et al. introduce a deep feature selection (DFS) method which selects features at the input layer of a deep neural network [LCW16]. A sparse one-to-one layer is added between the input layer and the first hidden layer of a neural network. The weights of this new layer are regularized with a sparse regularization term (the Elastic Net penalty), and only features with a non-zero weight are selected. Figure 2.15 depicts such a network. To the contrary of Elastic-Net Logistic Regression, Deep Feature Selection can model non-linear relationships between input features and the target variable by taking advantage of deep structures. Nevertheless, the learning of deep models is a non-convex optimization problem. This non-convexity comes with a set of issues and challenges (*e.g.*, stochastic behavior, convergence dependent on the learning rate). The Deep Feature Selection approach can also learn sparse linear models by using a shallow network without any hidden layer. Figure 2.16 depicts such a shallow network. The authors prove that this formulation is not equivalent to the classic Elastic-Net regression [LCW16].

Roy et. al [RMM15] also select features at the input level of a DNN. They propose to rank and select features based on their contribution to the activation potential of the first hidden layer, reflecting their likelihood of making neurons participate in the classification.

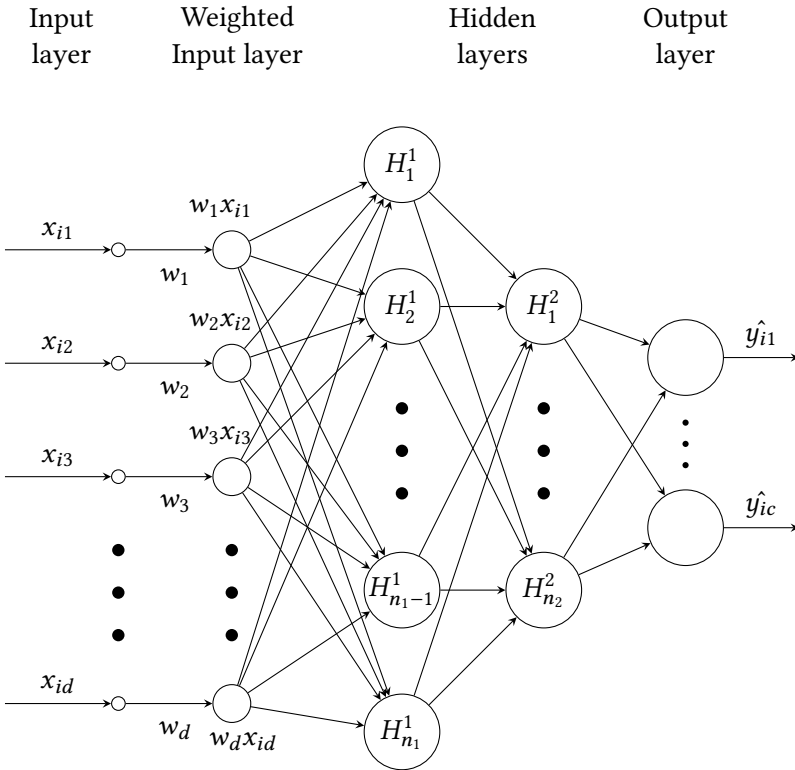


Figure 2.15: Feed-forward neural network with two hidden layers, preceded by a one-to-one layer with a linear activation function. Feature selection is performed by applying the elastic-net penalty to the one-to-one layer's weight vector w . Features with $w_f = 0$ have no impact on the network predictions and can thus be discarded.

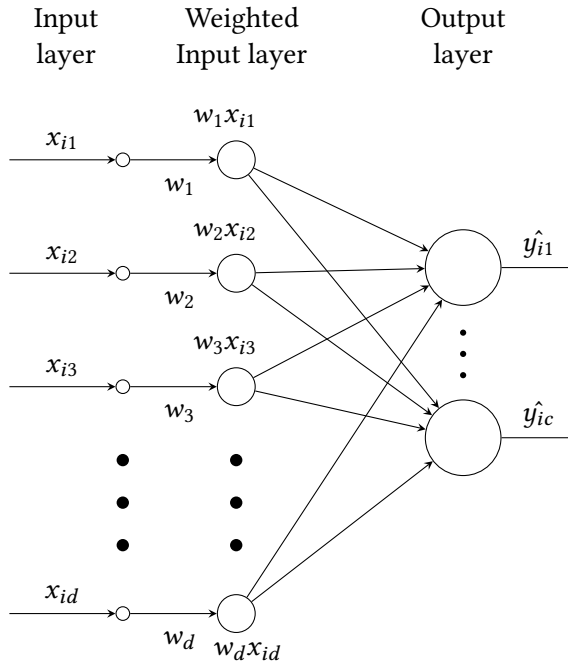


Figure 2.16: Shallow feed-forward neural network with no hidden layer, preceded by a one-to-one layer with a linear activation function. Feature selection is performed by applying the elastic-net penalty to the one-to-one layer’s weight vector w . Features with $w_f = 0$ have no impact on the network predictions and can thus be discarded.

2.3 Selection Stability

Feature selection started to be a hot topic in machine learning back in 1997 [BL97]. Yet, it was only in 2005 that the *stability* of feature selection was considered and defined for the first time [KPH05]. The stability of a feature selection algorithm is there defined as “the robustness of the feature preferences it produces to differences in training sets drawn from the same generating distribution”. Measuring selection stability is essential as stability directly relates to the selection algorithms’ trustworthiness. If minor changes in the training data largely influence the selected features, they are unlikely to accurately represent the true underlying processes at play. Selected features are often analyzed later by domain experts to gain further insight. Such experts will not invest resources into the analysis of some features if the algorithm that identifies them does not seem to be reliable. Those features might indeed not represent a real signal worth investigating. Consequently, domain experts often prefer a more stable feature selection algorithm over an unstable and slightly more accurate one [KPH07; SAV08a]. This is especially true in

the biomedical field, where reproducibility has proven to be a key challenge [Hai+13; Lee+13]. Selection stability is sometimes even argued to be equally important as predictive performance [Jur+08].

Since its first definition in 2005, selection stability has been considered in more and more papers when evaluating feature selection algorithms, notably in the context of microarray classification [Dav+06; Qiu+06; SK10; YHB11]. This new attention to selection stability fits well in the global *explainable IA* trend that attracts a lot of research effort nowadays [LPK21]. Designing more trustworthy feature selection and machine learning algorithms is indeed key to promoting machine learning in sensitive areas, such as in the biomedical domain.

Some works specifically study the reasons behind selection instability. Results show that it is mostly caused by the small sample/feature ratio [Ale13], noise in the data [Sha+12], an imbalanced target variable [Awa+12], or feature redundancy [SN10a]. While all of these reasons clearly play a role, the small sample/feature ratio and feature redundancy are likely the most important ones in a biomedical domain with typically several thousands, if not millions, of sometimes highly correlated features for only a few dozen or hundred samples. This is likely why stable feature selection is intrinsically hard in this domain and why existing techniques are still unsatisfactory.

Several authors have proposed different approaches to increase stability. For instance, instance-weighting for variance reduction [HY12] and ensemble methods for feature selection have been proposed [SAV08b; Abe+10] and generally increase selection stability. Stability selection [MB10] is a particular ensemble method which ultimately selects features with a selection frequency higher than a threshold π_{thr} for at least one regularisation parameter $\lambda \in \Lambda$. While these methods have been shown to increase selection stability, the gain they offer is still limited as they are not designed to search explicitly through a bi-dimensional (accuracy, stability) objective space. This notion of bi-objective optimization is covered in Chapter 3.

Figure 2.17 illustrates a classic protocol to evaluate the stability of a feature selection algorithm. The initial data set is perturbed⁴ to form M different data sets $\mathcal{D}_{1 \leq i \leq M}$. A feature subset $\mathcal{F}_{1 \leq i \leq M}$ is selected from each of these modified versions of the initial data set. A predictive model $\mathcal{P}_{1 \leq i \leq M}$ is then built on each feature subset and evaluated on some test examples. The pipeline depicted in Figure 2.17 has two non-necessarily competing objectives: 1) a measure of the performance of the predictive models built on the selected features and 2) a measure of the stability of the selected features, which is related to the soundness of the domain analysis. Possible additional quality criteria are minimal model size or sparsity. Instability arises when little

⁴It is here done by subsampling which is often used to measure such instability, but it could be any small perturbation.

agreement over the selected features occurs, *i.e.*, when the second objective is not met. This prevents a correct and sound interpretation of the selected features and strongly impacts their further validation by domain experts as it reduces their trust in the proposed features. Unlike optimizing the accuracy of predictive models, optimizing selection stability may look trivial since an algorithm always returning an arbitrary but fixed set of features would be stable by design. Yet, such an algorithm is not expected to select informative and predictive features and would thus fail to meet the first objective above. This illustrates that optimizing stability is only well-posed jointly with predictive accuracy, which can be defined as the average test accuracy of the M learned models (as in Figure 2.17).

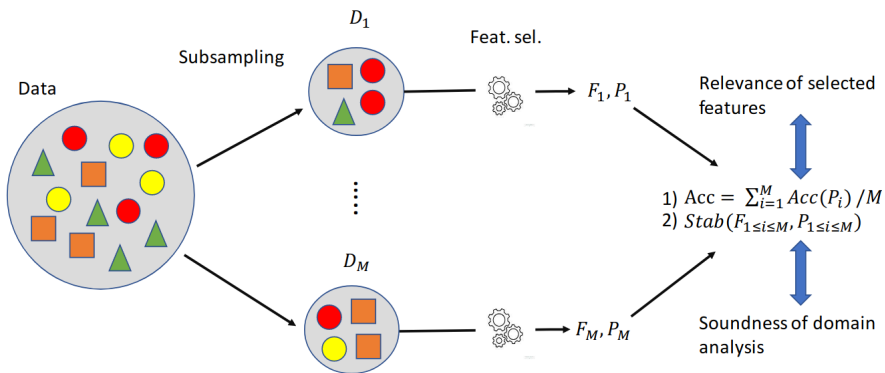


Figure 2.17: Illustration of the stability problem. The outcome is a trade-off between predictive accuracy and selection stability. The accuracy relates to the relevance of the selected features while stability is linked to the soundness of the domain analysis.

To assess the stability of a feature selection algorithm, one needs a *stability measure*. We refer to each row of Figure 2.17, *i.e.*, a selection step followed by the estimation of a predictive model on a given subsample, as a *selection run*. A stability measure aims to quantify the stability of M of these selection runs. Despite the simple and intuitive definition of stability, quantifying it has proven to be quite challenging.

Feature selection algorithms can produce either (a) a subset of selected features, (b) a partial or complete ranking of the features, or (c) a weight for each feature which typically represents the importance of this feature in a predictive model. Each type of feature selection requires dedicated stability measures. This document focuses on subset (a) and weighted (c) stability measures. Ranking measures (b) are indeed not well-suited to quantify the trust of domain experts in the selected features when those features are used to build predictive models in a second stage. Section 5.2 develops this argument

further. Still, we briefly consider the ranking measure proposed in [Jur+08].

The next three sections review subset stability measures, which quantify the stability of M feature subsets. Section 2.3.4 then presents some weight-based and ranking-based stability measures.

2.3.1 Similarity-based subset measures

The similarity approach defines stability as the average pairwise similarity between two distinct selection runs:

$$S_M = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \text{Sim}(\mathcal{F}_i, \mathcal{F}_j). \quad (2.34)$$

Many similarity-based stability measures have been proposed. They differ only by the choice of similarity function between two feature sets. For instance, Dunne et al. propose the relative Hamming distance [DCA02] while Kalousis et al. rather uses the Jaccard index [KPH07]. Other similarity-based measures include the Dice-Sørensen index [YDL08], the Ochiai index [ZRS08], the POG (Percentage of Overlapping Genes) and nPOG indices [Shi+06; Zha+09].

The Kuncheva Index Kuncheva analyzed some pre-existing stability measures and showed that they sometimes behave in an undesirable fashion [Kun07]. He formulated a set of three desirable properties, shown to be key for a sound comparison and interpretation of stability. A novel similarity-based measure, the Kuncheva index, satisfying these three properties is proposed in the same work:

$$\text{KI} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j| - \frac{k^2}{d}}{k - \frac{k^2}{d}}, \quad (2.35)$$

with k the number of features selected from the original d features. The measure essentially quantifies the fractional feature intersection $\frac{|\mathcal{F}_i \cap \mathcal{F}_j|}{k}$, adjusted by the terms $\frac{k^2}{d}$. These correction terms ensure that the Kuncheva index is *corrected for chance*, which is its main advantage over the contemporary competition. The incentive of this correction is that stability should not be impacted by feature intersections that occur by chance. More formally, the property of correction for chance imposes a particular behavior of a stability measure under the null model of feature selection H_0 . This null model refers to the situation where the features are selected uniformly at random. More specifically, a measure is said to be corrected for chance when its expected value under H_0 is constant (and usually set to 0). However, the Kuncheva index is only defined when the number of selected features is constant across all

selection runs. Several extensions of the Kuncheva index that overcome this issue have been proposed [LGV09; WKS12]. Unfortunately, they lose some desirable properties and are still not completely satisfactory [NSB17a].

2.3.2 Frequency-based subset measures

Frequency-based stability measures do not look at the pairwise similarity between selection runs. Instead, they are functions of the selection frequency of each feature across all selection runs. The output of a feature selection algorithm can be modelled as a binary matrix

$$\mathcal{Z} = \begin{pmatrix} z_{1,1} & z_{1,2} & \dots & z_{1,d} \\ z_{2,1} & z_{2,2} & \dots & z_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ z_{M,1} & z_{M,2} & \dots & z_{M,d} \end{pmatrix} \begin{matrix} \mathbf{z}_{1,-}, \text{ selected features in run 1} \\ \mathbf{z}_{2,-}, \text{ selected features in run 2} \\ \vdots \\ \mathbf{z}_{M,-}, \text{ selected features in run } M \end{matrix} \quad (2.36)$$

where $z_{i,f}$ is the Bernoulli variable associated to the selection of feature f in selection run i ($z_{i,f} = 1$ in the case of selection and $z_{i,f} = 0$ otherwise). The *selection frequency* of a feature f is then defined as the mean of the f th column of $\mathcal{Z}$: $p_f = \frac{1}{M} \sum_{i=1}^M z_{i,f}$. Frequency-based measures are function of these selection frequencies. Nogueira identifies 6 existing stability measures in this category [NSB17a]. For instance, Goh and Wong simply use the average selection frequency p_f of every feature selected at least once [GW16]. Davis et al. add a penalty term to account for the apparent stability increase that occurs with long gene signatures [Dav+06]. Other existing measures are based on the Shannon entropy [KKH07] and the Jensen-Shannon divergences [GA11], not to mention [SN10b; Lau+13].

Under such a profusion of different measures, it becomes difficult to justify the choice of a particular index and even more to compare the results of works based on different metrics. Furthermore, the large number of available measures can lead to publication bias (researchers may select the index that makes their algorithm look the most stable) [BS09]. In the hope of fixing this issue, a more recent work [NSB17a] lists and analyzes 15 different stability measures, most of them cited here-above. They are compared based on the satisfaction of 5 different properties that a stability measure should comply with. They also propose a novel and unifying frequency-based index. This index, which is extensively used in this work, is described in more detail in Section 2.3.3. A related and popular weight-based stability measure, which makes use of the sample Pearson's correlation coefficient [KPH07; NB16], is reviewed in Section 2.3.4.

2.3.3 The Nogueira Index

The novel stability index introduced by Nogueira et. al. [NSB17a] can be computed according to Equation (2.37)

$$\Phi = 1 - \frac{\frac{1}{d} \sum_{f=1}^d s_f^2}{\frac{\bar{k}}{d} * (1 - \frac{\bar{k}}{d})}, \quad (2.37)$$

with $\bar{k}$ the mean number of features selected from the original d features and $s_f^2 = \frac{M}{M-1} \hat{p}_f(1 - \hat{p}_f)$ the estimator of the variance of the selection of the f_{th} feature over the M selected subsets, where $\hat{p}_f$ is the fraction of times feature f has been selected among them. The Nogueira index essentially defines stability as 1 minus the average selection variance of all features. The bigger this average selection variance, the smaller the stability. The denominator $\frac{\bar{k}}{d} * (1 - \frac{\bar{k}}{d})$ normalizes the measure and corrects it for chance.

Desirable properties The Nogueira index is the only existing measure satisfying the five properties described in [NSB17a]:

- *Fully defined*: the measure is defined for every possible combinations of M feature subsets.
- *Strict monotonicity*: the measure strictly decreases with the selection variance.
- *Bounds*: the measure is bounded by constants.
- *Maximum stability $\Leftrightarrow$ deterministic selection*
- *Correction for chance*: under the null model of feature selection H_0 , the expected value of the measure is constant.

The *correction for chance* property states that Φ is constant in expectation (here set to 0) under H_0 (i.e., when the features are selected uniformly at random). As mentioned previously, the intuition behind this property is that a stability measure should not be influenced by the similarities between selected feature subsets that occur by chance. This is a requirement for the sound comparison of the stability of different approaches with different numbers of selected features. Indeed, the more features are selected, the more likely some feature intersections occur simply by chance. Measures that are not corrected for chance are biased towards large sets of selected features because they do not account for this phenomenon. Correction for chance is also required to compare stability values obtained on different data sets with a different number of input features d .

In addition to the *correction for chance* property, Φ is formally bounded by -1 and 1 and is asymptotically lower bounded by 0 as $M \rightarrow \infty$. This bound property is essential for the interpretability of the stability values. A value under 0 means that the selection is even less stable than random selection (which cannot happen when $M \rightarrow \infty$). A value of 1 indicates here perfect stability. Without lower or upper bounds, interpreting a specific stability value is difficult.

The *fully defined* property simply states that the measure is always defined. In particular, it is defined when the number of selected features varies across selection runs, which is not the case for the Kuncheva index.

The *maximum* property ensures that the stability value is maximal whenever the selection is deterministic, i.e., when the same feature set is selected across the M runs, and vice-versa.

Finally, the *strict monotonicity* property checks that the measure actually quantifies the original objective of stability. Nogueira et. al. bridged the gap between the two families of measures (similarity-based and frequency-based) by proving the following result:

$$\frac{2}{M(M-1)} \sum_{i=1}^{M-1} \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j| = \bar{k} - \sum_{f=1}^d s_f^2. \quad (2.38)$$

A stability index increasing with the size of the intersection between pairwise feature subsets can be re-formulated as another index, decreasing with the feature selection variance. In particular, the Nogueira index is equivalent to the Kuncheva index when the number of selected features k is constant across the M selected subsets. Still, it can be computed in $O(Md)$, whereas the Kuncheva index requires $O(M^2d)$. The *strict monotonicity* property also ensures that the measure strictly increases with the feature subsets intersection.

2.3.4 Weight and ranking based measures

This section reviews the most popular weight-based stability measure and an example of a ranking-based measure.

Pearson's Correlation This weight-based stability measure computes the average Pearson's correlation between feature weights of different selection runs:

$$\Phi_{\text{pears}} = \frac{2}{M(M-1)} \sum_{i=1}^{M-1} \sum_{j=i+1}^M \rho_{i,j}, \quad (2.39)$$

where

$$\rho_{i,j} = \frac{\sum_{f=1}^d (w_{f,i} - \mu_i)(w_{f,j} - \mu_j)}{\sqrt{\sum_{f=1}^d (w_{f,i} - \mu_i)^2} * \sqrt{\sum_{f=1}^d (w_{f,j} - \mu_j)^2}},$$

with $w_{f,i}$ the weight, or score, associated to feature f in selection run i and μ_i the average feature weight in this run. If these weights are either 0 or 1 (indicating the selection of the feature) and if the number of non-zero weights is constant across selection runs, then Φ_{pears} is equivalent to the Kuncheva index [NB16]. By extension, it is also equivalent to Φ in this particular setting.

The Canberra Distance Between Partial Rankings The Canberra distance, with location parameter k , evaluates the stability of partial feature rankings (of size k) and is defined as follows [Jur+08]:

$$\Phi_{\text{can}}^k = 1 - \frac{1}{\chi} \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\min(r_i^f, k+1) - \min(r_j^f, k+1)|}{\min(r_i^f, k+1) + \min(r_j^f, k+1)}, \quad (2.40)$$

with r_i^f the rank of feature f in ranking i and $\chi = \frac{(k+1)(2d-k)}{d} \times \log(4) - \frac{2kd+3d-k-k^2}{d}$ the term that approximately corrects the measure for chance. This measure naturally penalizes more variability that occurs at the top of the ranking.

The index Φ_{can}^k is used as a representative of ranking-based measures in Chapter 5. We refer the reader to [UPC20; Jur+08; NSB17b; KV10] for other instances of ranking-based measures.

A Bi-Objective Evaluation Framework

3

This chapter introduces the notion of joint optimization of selection stability and predictive performance. Many papers consider selection stability as a side-objective of prediction performance. Some acknowledge the existence of an (accuracy, stability) Pareto-front, from which domain experts can choose a particular trade-off [NSB17a; Sec+19]. Yet, they do not evaluate the *aggregated* performance of feature selection and cannot compare Pareto-fronts to each other (unless one is dominated by the other). In this chapter, we fill this gap and propose an evaluation framework that estimates the efficiency of methods balancing between classification performance and selection stability. The chapter is organized as follows. It first reviews the well-known hypervolume (HV) performance metric in Section 3.2. It then defines a generic evaluation framework based on the notion of satisfaction of domain experts, from which the hypervolume can be re-derived with a specific set of parameters (Section 3.3). However, we show that the hypervolume is unlikely to measure the satisfaction of domain experts adequately, and a novel alternative is proposed, namely the expected satisfaction metric. Statistical tools for inference about this novel metric are gradually built in Sections 3.4. In particular, confidence intervals on the expected satisfaction are built, along with hypothesis tests that quantify the statistical significance of observed differences between Pareto-fronts.

3.1 The objective-space

Classification performance Through the first chapters of this thesis, the phrases ‘predictive accuracy’ and ‘classification performance’, among others, were used without specifying how such a performance is actually measured. The most broadly used classification metric is undoubtedly the accuracy, also known as the correct classification rate. It is the ratio of the number of correct predictions to the number of test examples:

$$\text{Acc} = \frac{1}{n} \sum_{i=1}^n \hat{y}_i == y_i. \quad (3.1)$$

In the presence of a strong class imbalance, the accuracy is not an adequate performance indicator. Indeed, a classifier always predicting the majority class (the class with the highest class prior θ_{maj}) has an accuracy of $\theta_{\text{maj}} \geq 0.5$. A better alternative to the accuracy which correctly deals with class imbalance is the balanced classification rate, also known as the balanced accuracy. It is the average of the ratio of positively labeled examples that are correctly classified as positive (also known as sensitivity) and the ratio of the negatively labeled examples that are correctly classified as negative (also known as specificity). Let n_{ij} be the number of examples with true label i predicted as belonging to class j . Let n_i be the total number of examples with true label i . The balanced accuracy for a binary prediction task is

$$a_b = \frac{1}{2} \left(\frac{n_{++}}{n_+} + \frac{n_{--}}{n_-} \right). \quad (3.2)$$

It can be generalized to C classes intuitively:

$$a_b = \frac{1}{C} \sum_{c=1}^C \frac{n_{cc}}{n_c}. \quad (3.3)$$

A classifier that always predicts the same class has a balanced accuracy of $\frac{1}{C}$, no matter how the class priors are distributed.

Selection Stability As in most papers, we use a subset stability measure, here the Nogueira index Φ introduced in Section 2.3.3.

The objective space We consider binary classification tasks and evaluate the performance of predictive models using the balanced accuracy. In the rest of this document, the mainstream term accuracy denotes this balanced accuracy unless specified otherwise. In this context, we opt to restrict the objective space to $P = \{(a_b, \Phi) : 0.5 \leq a_b \leq 1, 0 \leq \Phi \leq 1\}$. A balanced accuracy of 0.5 is trivially achievable by always predicting either one of the two classes. The value 0.5 can be considered a lower bound. A balanced accuracy of $\theta < 0.5$ can be reverted to $(1-\theta) > 0.5$ by inverting all predictions anyway. Furthermore, the careful reader may remember that the stability measure Φ formally lies in the $[-1, 1]$ interval. However, as $\Phi = 0$ corresponds to the stability of a uniformly random selection, we argue that the only interesting part of the stability spectrum is in fact $[0, 1]$.

Pareto optimality A pair (a_{b_1}, Φ_1) dominates another pair (a_{b_2}, Φ_2) if and only if $a_{b_1} \geq a_{b_2}$, $\Phi_1 \geq \Phi_2$, and at least one of the inequalities is strict ($>$). A given selection method m is able to generate some pairs P_m in the space of all possible pairs P . From the set of generated pairs P_m , the set of pairs that

are not dominated by any other pair can be found. This set, called the Pareto set, defines a subspace where no point dominates any other point. A domain expert would then choose his favorite pair based on his personal preference for classification performance and feature selection stability. In the following, we detail the two bi-objective performance metrics that are considered in this thesis. These metrics evaluate the performance of a Pareto-front.

3.2 The Hypervolume

As first performance metric, we use the well-known hypervolume indicator [ZT99], also known as S-metric. This indicator is defined as the size of the objective space that is dominated by at least one point of the Pareto set. It has the convenient property that whenever a Pareto set dominates another, the hypervolume of the former is greater. This metric is the most widely used in multi-objective optimization [RVB15].

The hypervolume (HV) over P is

$$HV_P = 2 \int_{0.5}^1 \int_0^1 f_P(a_b, \Phi) da_b d\Phi, \quad (3.4)$$

with f_P the attainment function, which is equal to 1 if (a_b, Φ) is dominated and 0 otherwise. The normalization term 2 ensures that the hypervolume lies in the interval $[0, 1]$. To simplify further mathematical expressions, we pose the normalized balanced accuracy as $\tilde{a}_b = 2(a_b - 0.5)$, such that

$$HV_P = \int_0^1 \int_0^1 f_P(\tilde{a}_b, \Phi) d\tilde{a}_b d\Phi. \quad (3.5)$$

An illustration of the hypervolume indicator can be seen in Figure 3.1. This figure depicts two Pareto fronts supposedly obtained via two feature selection methods. We display the non re-normalized balanced accuracy a_b when plotting the bi-objective space for interpretability reasons. The blue front starts from the left with a higher (balanced) accuracy. It thus gains some area over the red front. Nonetheless, the red front reaches higher stabilities without dropping the accuracy as much as the blue one. Overall, the red front has a larger hypervolume. Note that this hypervolume is also equal to the fraction of the total area that is dominated by the method, or 1 minus the fraction of area that dominates the method.

Some authors propose to give different weights to different portions of the objective space [ZBT07]. Applied to our context, this weighted hypervolume can be computed via the weighted integral

$$HV_P^w = \int_0^1 \int_0^1 w(\tilde{a}_b, \Phi) f_P(\tilde{a}_b, \Phi) d\tilde{a}_b d\Phi, \quad (3.6)$$

with w the weighting function. To preserve the $[0 - 1]$ bounds, w has to be normalized such that its integral over the objective space is 1. For example, the normalized weighting function $w_A(\tilde{a}_b, \Phi) \triangleq A \frac{e^{A \cdot \tilde{a}_b}}{e^A - 1}$ gives a higher weight to the portions of the space where the accuracy is high. In the example of Figure 3.1, the blue front actually outperforms the red one for $A > 2.38$. Without prior knowledge about the preference of experts, our methods are evaluated with $w(\tilde{a}_b, \Phi) = 1$ ($\Leftrightarrow A \rightarrow 0$, as $\lim_{A \rightarrow 0} \frac{A}{e^A - 1} = 1$) but the proposed evaluation framework allows for more, for instance, if domain experts are particularly interested in some parts of the objective space.

The hypervolume indicator rewards points with both good accuracy and good stability. Points with only one of the above contribute much less to the metric. At the limit, a feature selection algorithm with *perfect* accuracy but zero stability has a null hypervolume, whatever the value of the weighting parameter A . This could be seen as a disadvantage of the metric, as a good accuracy is still definitely better than nothing, yet the hypervolume does not reward it when stability is very poor. We define another performance metric with a different behavior in the next section.

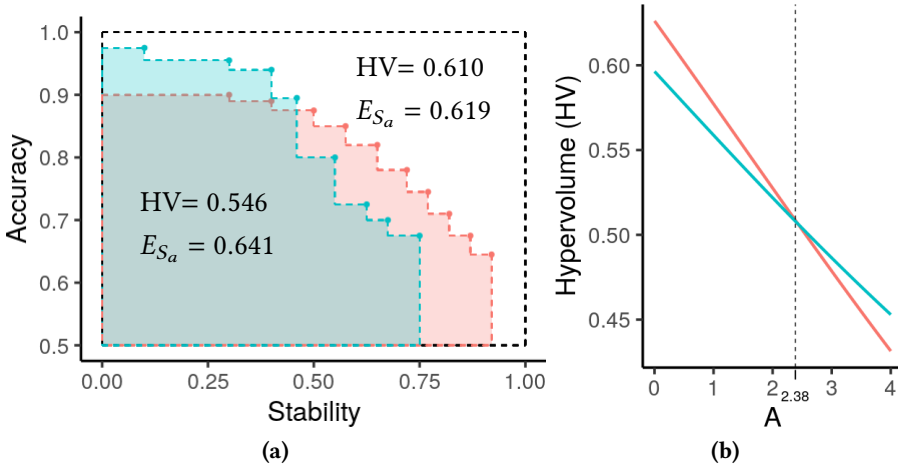


Figure 3.1: Illustration of the hypervolume indicator (HV). It represents the area that is dominated by at least one point generated by the method (a). Dependence of the weighted hypervolume HV_p^w to the constant A (b).

3.3 Expected satisfaction

In practice, given a Pareto set Pa , a domain expert would choose a single point from the set based on his personal preferences. We here formalize this notion of personal preference under the form of a *satisfaction* function S_a that experts wish to maximize when opting for a particular Pareto point. Given a

satisfaction function S_a , the chosen Pareto point is then

$$(\tilde{a}_b^*, \Phi^*) = \underset{(\tilde{a}_b, \Phi) \in Pa}{\operatorname{argmax}} S_a(\tilde{a}_b, \Phi) \quad (3.7)$$

and the expert satisfaction is simply

$$S_a(\tilde{a}_b^*, \Phi^*) = \max_{(\tilde{a}_b, \Phi) \in Pa} S_a(\tilde{a}_b, \Phi).$$

As performance metric, we use the *expected satisfaction*

$$E_{S_a} = \int_{S_a} p(S_a) \max_{(\tilde{a}_b, \Phi) \in Pa} S_a(\tilde{a}_b, \Phi) dS_a, \quad (3.8)$$

with $p(S_a)$ the probability that a domain expert has the satisfaction function S_a . The expected satisfaction E_{S_a} computes the average satisfaction of a domain expert after selecting his favorite compromise from the evaluated Pareto-front.

This evaluation framework is generic, but to actually estimate E_{S_a} , one has to restrict the space of possible functions S_a . We make our choice based on the following four axioms:

1. Different domain experts have different preferences for accuracy and stability. Their satisfaction depends on some set of parameters γ , whose values depend on each expert:

$$S_a = S_a(\tilde{a}_b, \Phi, \gamma).$$

2. The point $(\tilde{a}_b = 0, \Phi = 1)$ is trivially achievable by always returning an arbitrary but fixed subset of features ($\Phi = 1$) and always predicting the majority class ($\tilde{a}_b = 0$). Such a point should not provide any satisfaction to domain experts. In other words, stability without accuracy is meaningless.

$$S_a(\tilde{a}_b = 0, \Phi = 1) = 0.$$

3. S_a is an increasing function of $\tilde{a}_b$ and Φ :

$$\frac{\partial S_a}{\partial \tilde{a}_b} \geq 0, \quad \frac{\partial S_a}{\partial \Phi} \geq 0.$$

4. The points $(\tilde{a}_b > 0, \Phi = 0)$ are **not** meaningless. Accuracy without stability is not trivial:

$$S_a(\tilde{a}_b > 0, \Phi = 0) > 0.$$

For normalization purposes, we also impose $S_a(\tilde{a}_b = 1, \Phi = 1) = 1$, such that S_a is bounded between 0 and 1. The *expected satisfaction* then becomes

$$E_{S_a} = \int_{\gamma} p(\gamma) \max_{(\tilde{a}_b, \Phi) \in Pa} S_a(\tilde{a}_b, \Phi, \gamma) d\gamma, \quad (3.9)$$

with p the probability density function of γ . We now assume that the satisfaction functions are of the simple form

$$S_a(\tilde{a}_b, \Phi, \gamma) = \gamma \tilde{a}_b + (1 - \gamma) \Phi \sqrt{\tilde{a}_b}, \quad (3.10)$$

with a single parameter $\gamma \in]0, 1]$. This function satisfies the four above axioms¹ and is bounded between 0 and 1. The closer γ is to 1, the more weight is put on the accuracy. The $\sqrt{\tilde{a}_b}$ term ensures that axiom number 2 is respected, *i.e.*, that stability without any accuracy is never promoted. The contour lines of S_a are depicted on Figure 3.2 for three specific values of γ . Such contour lines represent a path of constant satisfaction for domain experts (they do not prefer any point of the line over the others). Without any other knowledge, we further assume that every value of γ is equiprobable, such that $p(\gamma) = 1$ if $0 < \gamma \leq 1$, and 0 otherwise. The expected satisfaction can then be written as

$$E_{S_a} = \int_0^1 \max_{(\tilde{a}_b, \Phi) \in Pa} \gamma \tilde{a}_b + (1 - \gamma) \Phi \sqrt{\tilde{a}_b} d\gamma. \quad (3.11)$$

Under this formulation, E_{S_a} is bounded in $[0, 1]$. As for the hypervolume, the expected satisfaction can be weighted by assuming a different probability distribution of γ .

Unlike for the hypervolume, the accuracy edge of the bi-objective space contributes significantly to the expected satisfaction. In the limit case of perfect accuracy and zero stability, the expected satisfaction E_{S_a} is equal to $\int_0^1 \gamma d\gamma = 0.5$. Yet, it can be shown that some Pareto points are not eligible to maximize the function $\gamma \tilde{a}_b + (1 - \gamma) \Phi \sqrt{\tilde{a}_b}$, for any $0 < \gamma \leq 1$. These Pareto points never contribute to the metric. As a consequence, it is possible that $Pa_x \subset Pa_y$ and that $E_{S_a}(x) = E_{S_a}(y)$. In other words, the front y has the set of Pareto points of the front x plus some others, which cover more area, but the expected satisfaction metric does not reward this. The expected satisfaction values are also displayed in Figure 3.1. In this handcrafted example, the metrics do not agree.

¹To meet axiom number 4 for all γ values, we forbid γ to be strictly equal to 0. This is a technicality and it does not impact the value of E_{S_a} , defined in Equation 3.11.

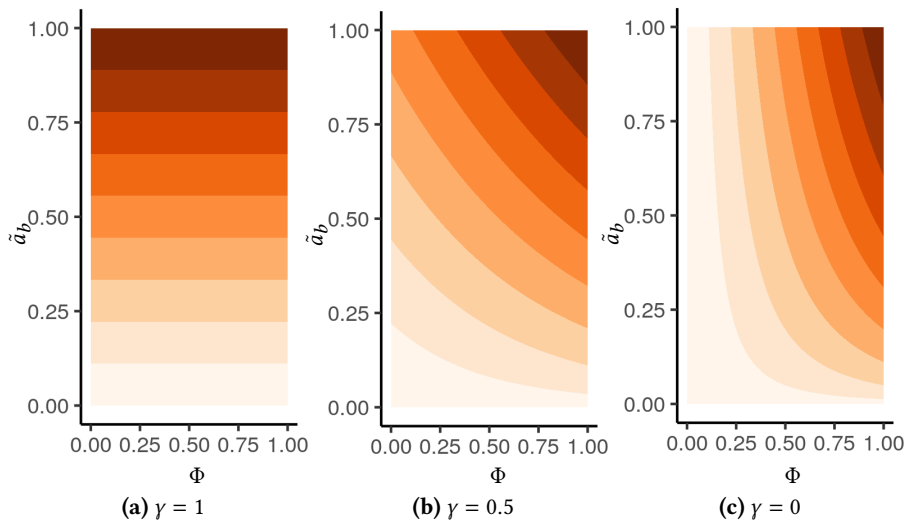


Figure 3.2: Contour plots of $S_a(\tilde{a}_b, \Phi, \gamma)$ for $\gamma = 1$ (a), $\gamma = 0.5$ (b), and $\gamma = 0$ (c). When $\gamma = 1$, stability is completely disregarded and $S_a = \tilde{a}_b$. The contour lines are then horizontal. When $\gamma = 0.5$, the contour lines are diagonal with slopes close to 1 at the top, slopes that decrease at the bottom when $\tilde{a}_b \rightarrow 0$ such that stability without any accuracy is never promoted. When $\gamma = 0$, the focus on stability is maximal, and the contour lines are much more vertical.

This framework can be used to derive the (unweighted) hypervolume indicator by posing the following satisfaction function

$$S_a^{hv}(\tilde{a}_b, \Phi, \gamma) = \begin{cases} \Phi & \text{if } \tilde{a}_b \geq \gamma \\ 0 & \text{otherwise.} \end{cases} \quad (3.12)$$

Then,

$$\begin{aligned} E_{S_a} &= \int_0^1 \max_{(\tilde{a}_b, \Phi) \in Pa} S_a^{hv}(\tilde{a}_b, \Phi, \gamma) d\gamma = \int_0^1 \int_0^1 f_P(\tilde{a}_b = \gamma, \Phi) d\Phi d\gamma \\ &= \int_0^1 \int_0^1 f_P(\tilde{a}_b, \Phi) d\tilde{a}_b d\Phi = HV_P. \end{aligned} \quad (3.13)$$

Note that this satisfaction function does not satisfy the fourth axiom. The same result can be obtained with

$$S_a^{hv}(\tilde{a}_b, \Phi, \gamma) = \begin{cases} \tilde{a}_b & \text{if } \Phi \geq 1 - \gamma \\ 0 & \text{otherwise} \end{cases}, \quad (3.14)$$

which also fails to meet axiom number 4.

3.4 Evaluation Protocol

Sections 3.2 and 3.3 describe how to compute the hypervolume and expected satisfaction aggregated performance metrics from a given set of Pareto-optimal points. This section details how these points are actually estimated from data. Furthermore, it develops statistical tools on both metrics. For instance, in Figure 3.1, such tools could be used to assess if the observed differences between the two Pareto-fronts are *statistically significant*.

We use the evaluation protocol depicted in Figure 3.3. Concretely, M bootstrap samples are drawn from the original data X . Feature selection is performed, and a predictive model is learned on each of these M bootstraps. Let O_Λ denote a particular feature selection algorithm with the meta-parameter set Λ . The two quantities to be estimated are

1. The average test balanced accuracy (on new, unseen data) ${}_b\text{Acc}_{O_\Lambda}^X$ of predictive models learned on bootstrap samples of X :

$${}_b\text{Acc}_{O_\Lambda}^X \triangleq E_{X_i} [{}_b\text{Acc}_{O_\Lambda}^{X_i}(\mathcal{P}_X)], \quad (3.15)$$

where X_i is the bootstrap sample on which O_Λ is trained in run i , and where $\mathcal{P}_X$ denotes the whole population of examples drawn from the

same distribution as X . An estimate of ${}_b\text{Acc}_{O_\Lambda}^X$ is the average accuracy of the M models evaluated on each model's out-of-bag examples:

$${}_b\hat{\text{Acc}}_{O_\Lambda}^X = \frac{1}{M} \sum_{i=1}^M {}_b\text{Acc}_{O_\Lambda}^{X_i}(\text{OOB}_i). \quad (3.16)$$

The quantity ${}_b\text{Acc}_{O_\Lambda}^X$ is the expected balanced accuracy of the predictive models on which domain analysis is performed. It will remain unknown, but inference can be made on it using statistical tools developed in Section 3.4.2.

2. The stability $\Phi_{O_\Lambda}^X$ of O_Λ when applied on bootstrap samples of X :

$$\Phi_{O_\Lambda}^X \triangleq \lim_{M \rightarrow \infty} \Phi(\mathcal{F}_{1..M}). \quad (3.17)$$

An estimate of $\Phi_{O_\Lambda}^X$ is given by measuring the stability of the M feature sets

$$\hat{\Phi}_{O_\Lambda}^X = \Phi(\mathcal{F}_{1..M}).$$

The asymptotic stability $\Phi_{O_\Lambda}^X$ of O_Λ represents the true confidence of O_Λ in the selected features on which domain analysis is performed. It will remain unknown, but inference can be made on it using statistical tools developed in Section 3.4.1.

We define the *hidden* satisfaction as

$$S_a^h = S_a({}_b\text{Acc}_{O_\Lambda}^X, \Phi_{O_\Lambda}^X). \quad (3.18)$$

As for ${}_b\text{Acc}_{O_\Lambda}^X$ and $\Phi_{O_\Lambda}^X$, this hidden satisfaction remains unknown. The rest of Section 3.4 is dedicated to building statistical tools for inference on S_a^h and its expected value $E_{S_a^h}$.

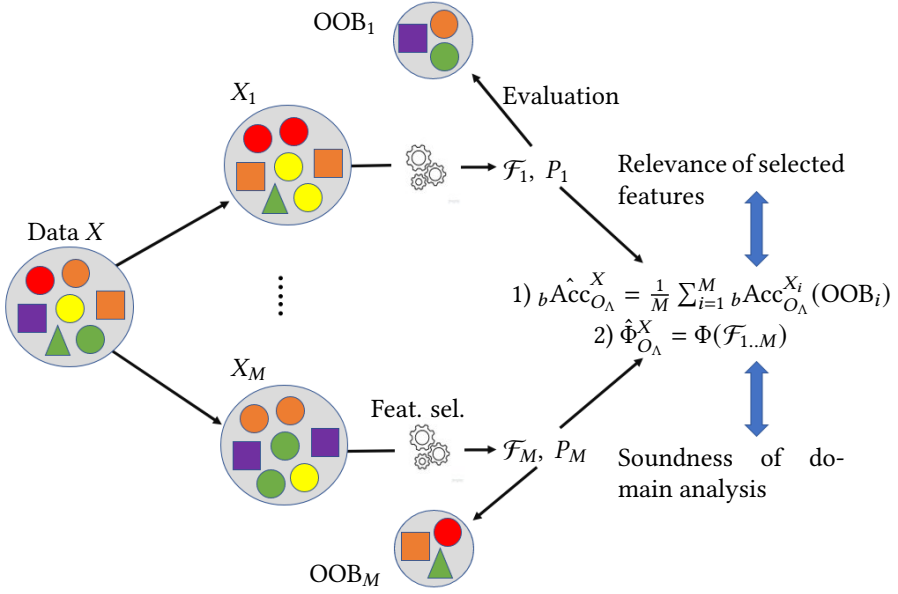


Figure 3.3: Evaluation protocol: M bootstrap samples are drawn from the original data X . Feature selection is performed on each bootstrap, and the resulting predictive model is evaluated on the out-of-bag examples. An estimate of the test accuracy of the algorithm O_Λ is obtained by averaging each model's accuracy on its out-of-bag examples. The stability measure Φ is used to compute the stability estimate of O_Λ .

3.4.1 Statistical tools of stability

This section summarizes the existing statistical tools on the stability $\Phi_{O_\Lambda}^X$ (Equation 3.17), developed by Nogueira et. al. [NSB17a]. For simplicity of notations, we omit the superscript X and the subscript O_Λ and note the true stability as Φ and its estimated value $\hat{\Phi}$.

Nogueira et. al. show that the estimated stability $\hat{\Phi}$ weakly converges to a normal distribution as $M \rightarrow \infty$:

$$\frac{\hat{\Phi} - \Phi}{\sqrt{v(\hat{\Phi})}} \rightarrow \mathcal{N}(0, 1), \quad v(\hat{\Phi}) = \frac{4}{M^2} \sum_{i=1}^M (\hat{\Phi}_{(i)} - \hat{\Phi}_{(\cdot)})^2, \quad (3.19)$$

with $v(\Phi)$ an estimate of the variance of $\hat{\Phi}$, i the index of the selection run,

$$\hat{\Phi}_{(i)} = \frac{1}{\bar{k}(1 - \frac{\bar{k}}{d})} \left[\frac{1}{d} \sum_{f=1}^d z_{i,f} p_f - \frac{k_i \bar{k}}{d^2} + \frac{\hat{\Phi}}{2} \left(\frac{2\bar{k}k_i}{d^2} - \frac{k_i}{d} - \frac{\bar{k}}{d} + 1 \right) \right], \quad (3.20)$$

where $z_{i,f}$ is a binary indicator variable equal to 1 if feature f is selected in

run i , and $\hat{\Phi}_{(.)} = \frac{1}{M} \sum_{i=1}^M \hat{\Phi}_{(i)}$. A $(1 - \alpha)\%$ confidence interval for Φ is then

$$[\hat{\Phi} - z_{1-\frac{\alpha}{2}}^* \sqrt{v(\Phi)}, \hat{\Phi} + z_{1-\frac{\alpha}{2}}^* \sqrt{v(\Phi)}], \quad (3.21)$$

where $z_{1-\frac{\alpha}{2}}^*$ is the inverse cumulative of a standard normal distribution at $1 - \frac{\alpha}{2}$.

To assess whether stability is statistically different from a given value or from the stability of another selection algorithm, we rely on *hypothesis testing*.

Is the true stability Φ greater than a given value Φ_0 ? In this statistical analysis, we assume the *null hypothesis* H_0 that $\Phi = \Phi_0$ and assess if our data can reject H_0 with confidence level $(1 - \alpha)$. The variable α is the accepted risk of making a type I error, *i.e.*, the risk of rejecting H_0 even though it is true. The alternative hypothesis H_1 is here $\Phi > \Phi_0$. The test statistic is

$$T_{\Phi_0} = \frac{\hat{\Phi} - \Phi_0}{\sqrt{v(\hat{\Phi})}}. \quad (3.22)$$

If $T_{\Phi_0} \geq z_{1-\alpha}^*$, one can reject the null hypothesis H_0 and conclude that H_1 is true, *i.e.*, that $\Phi > \Phi_0$. The *p-value* of the statistical test is the smallest α value for which the null hypothesis H_0 can be rejected. Note that this hypothesis testing is equivalent to constructing a one-sided confidence interval on Φ and inspecting if Φ_0 is in the interval or not. Indeed, such a one-sided confidence interval for Φ is

$$[\hat{\Phi} - z_{1-\alpha}^* \sqrt{v(\hat{\Phi})}, 1]. \quad (3.23)$$

One can easily check that Φ_0 is in the interval if and only if $T_{\Phi_0} \geq z_{1-\alpha}^*$.

Are the stabilities of two selection algorithms equal ? In this case, the null hypothesis H_0 is $\Phi_1 = \Phi_2$ and the alternative H_1 is $\Phi_1 \neq \Phi_2$. Assuming independence,

$$\hat{\Phi}_2 - \hat{\Phi}_1 \rightarrow \mathcal{N}(\Phi_2 - \Phi_1, v(\hat{\Phi}_2) + v(\hat{\Phi}_1)) \stackrel{H_0}{=} \mathcal{N}(0, v(\hat{\Phi}_2) + v(\hat{\Phi}_1)). \quad (3.24)$$

The statistic is now

$$T_{\Phi_{21}} = \frac{\hat{\Phi}_2 - \hat{\Phi}_1}{\sqrt{v(\hat{\Phi}_2) + v(\hat{\Phi}_1)}}. \quad (3.25)$$

If $|T_{\Phi_{21}}| \geq z_{1-\frac{\alpha}{2}}^*$, one can reject the null hypothesis H_0 and conclude that the stabilities are statistically different. Once again, this methodology is equivalent to constructing a $(1 - \alpha)\%$ confidence interval on $\Phi_2 - \Phi_1$ and checking if 0 belongs to it.

3.4.2 Statistical tools of accuracy

This section reviews existing and introduce new statistical tools on $b\text{Acc}_{O_\Lambda}^X$.

As the out-of-bag examples of the M predictive models overlap, the test accuracies $b\text{Acc}_{O_\Lambda}^{X_i}$ (OOB_i) are not independent. Special care must be given to account for this correlation when designing confidence intervals for their mean $b\text{Acc}_{O_\Lambda}^X$. Nadeau and Bengio show that

$$\frac{\hat{\mu}_\theta - \mu_\theta}{\sqrt{(\frac{1}{M} + \frac{\rho}{1-\rho})S_\theta^2}} \sim t_{M-1}(0, 1), \quad (3.26)$$

with t_{M-1} the Student's t distribution with $M - 1$ degrees of freedom, θ a generic estimator, $\hat{\mu}_\theta$ the sample mean of estimator θ over M runs, S_θ^2 its sample variance and ρ the correlation between the random variables θ_i [NB99]. This result strictly holds whenever the variables θ_i are normally distributed. This is true, for instance, when θ_i measures the standard (non-balanced) accuracy of the model learned in run i (provided that the number of out-of-bag examples exceeds 30). When the balanced accuracy is considered, *i.e.*, when $\theta_i = b\text{Acc}_{O_\Lambda}^{X_i}$ (OOB_i), normality can no longer be proven as the class priors in the out-of-bag test sets are also random variables. Nonetheless, we experimentally observe, as detailed in the rest of this section, that statistical analysis based on Equation 3.26 works nicely also when estimating the balanced accuracy. Particularizing Equation 3.26 to the balanced accuracy,

$$\frac{b\hat{\text{Acc}}_{O_\Lambda}^X - b\text{Acc}_{O_\Lambda}^X}{\sqrt{(\frac{1}{M} + \frac{\rho}{1-\rho})S_{\text{Acc}}^2}} \sim t_{M-1}(0, 1), \quad (3.27)$$

with ρ the correlation between the out-of-bag accuracies of the M models. A $(1 - \alpha)\%$ confidence interval for $b\text{Acc}_{O_\Lambda}^X$ is then

$$[b\hat{\text{Acc}}_{O_\Lambda}^X \pm t_{M-1, 1-\frac{\alpha}{2}}^* \sqrt{(\frac{1}{M} + \frac{\rho}{1-\rho})S_{\text{Acc}}^2}]. \quad (3.28)$$

The resampled t -test technique boldly assumes independence and puts ρ to 0, which induces liberal inference that worsens as M increases [Die98]. Nadeau and Bengio propose the corrected resampled technique where they (somewhat arbitrarily) set ρ to $\rho_0 = \frac{n_2}{n}$, with n_2 the number of test examples in each dependent test set and n the total number of examples. The fraction $\frac{n_2}{n}$ is thus equal to the expected fraction of examples shared by two distinct test sets. Applying this methodology to our bootstrap protocol, we get $\rho_0 = 1 - 0.632 = 0.368$. They propose this fixed correlation value ρ_0 as $\rho = \rho_0$ in the idealized scenario where the prediction of a model on a test example x_i

only depends on x_i , not on the train examples on which the model is built. In real-life scenarios, inference would be conservative when $\rho > \rho_0$ and liberal when $\rho < \rho_0$ [NB99].

Even though ρ_0 is computed in the scenario where there is no training set dependence, the work of Nadeau and Bengio aims to bound Acc_{O_Λ} , the average test accuracy of predictive models built on any data set from the same generative distribution as X (and of fixed size, depending on the protocol). Setting ρ to ρ_0 to bound $b\text{Acc}_{O_\Lambda}^X$ instead of $b\text{Acc}_{O_\Lambda}$ leads to too conservative inference. We here use an adaptation of [NB99] which leads to more powerful inference. Concretely, we set ρ to $\rho_0 \hat{\rho}_s$, with $\hat{\rho}_s$ the estimated correlation (weighted by class priors) between the (correctness of) predictions of the M models when evaluated on their shared test examples². More formally, let the random variable

$${}_j C_i^m(O_\Lambda) = \begin{cases} 1 & \text{if example } i \text{ of class } j \text{ is predicted correctly by } O_\Lambda \text{ in run } m \\ 0 & \text{otherwise.} \end{cases} \quad (3.29)$$

We then write (motivated by simple developments of the correlation ρ)

$$\rho_s \triangleq \frac{(\frac{n}{n_+})^2 \text{Cov}(+C_i^m, +C_i^{m'}) + (\frac{n}{n_-})^2 \text{Cov}(-C_i^m, -C_i^{m'})}{(\frac{n}{n_+})^2 \text{Var}(+C_i^m) + (\frac{n}{n_-})^2 \text{Var}(-C_i^m)}, \quad (3.30)$$

with $m \neq m'$ and i the index of an out-of-bag example in both runs. Using the data at hand, we get the estimate $\hat{\rho}_s$. In the base scenario where there is no training set dependence, $\hat{\rho}_s = 1$, which makes ρ equal to $\rho_0 = \frac{n_2}{n}$, and our methodology is equivalent to the corrected resampled t-test.

The estimator $\hat{\rho}_s$ has its own variance, and underestimating it can make the statistical analysis too liberal, especially for high confidence levels $(1 - \alpha)$. To alleviate this effect, we rather compute $B = \frac{1}{1 - \alpha}$ correlation values $\hat{\rho}_s^b$ on bootstrap samples of X and set

$$\hat{\rho}_s = \max_{1 \leq b \leq B} (\hat{\rho}_s^b).$$

Note that this does not require additional model estimations, we just randomly omit some examples and repeat others in the correlation estimation.

Figure 3.4 compares the coverage probabilities³ of the original confidence intervals of Nadeau and Bengio with our refined approach, applied to bound $b\text{Acc}_{O_\Lambda}^X$. This is evaluated on 9 data sets (introduced later in Table 4.1 and

²Such a methodology is not applicable when bounding $b\text{Acc}_{O_\Lambda}$. It would require measuring the correlation of predictions of common examples from models built on independent data, which is not practical.

³The coverage probability of a confidence interval on a variable x is the proportion of the time the confidence interval indeed contains the true value of x .

Table 5.2) with various numbers of examples n , numbers of features d , and class imbalances. For each data set, we use $M \in \{5, 15, 30, 50, 100\}$. As these data sets are finite and often quite small, measuring the true accuracy is impossible. We use the `mvtnorm` R package to generate unlimited (normally distributed) data with the same feature means, covariance matrices, and class imbalances as the original data sets. We consider an observed coverage probability p as *overly conservative* when $p > 1 - \frac{\alpha}{2}$. As liberal inference is often regarded as less desirable than conservative one, we consider p as *overly liberal* when $p < 1 - \frac{3}{2}\alpha$ instead of the reciprocal $p < 1 - 2\alpha$. Clearly, while the original confidence intervals of Nadeau and Bengio are largely overly conservative, our refined methodology is satisfactory. In this set of experiments, $\hat{\rho}_s$ takes values mostly between 0.4 and 0.85. The applied correction $\frac{\rho}{1-\rho}$ is thus from ~ 1.25 to ~ 3.5 times smaller than the correction applied by Nadeau and Bengio. Note that the (non-corrected) resampled t-test confidence intervals often have coverage probabilities smaller than 0.3 for $M = 100$, which makes it much too liberal.

Hypothesis testing can naturally be performed to assess if differences between estimated accuracies are significant.

Is the true accuracy ${}_b\text{Acc}_{O_\Lambda}^X$ greater than a given value ${}_b\text{Acc}_0$? The null hypothesis H_0 is here ${}_b\text{Acc}_{O_\Lambda}^X = {}_b\text{Acc}_0$ and the alternative H_1 is ${}_b\text{Acc}_{O_\Lambda}^X > {}_b\text{Acc}_0$. The test statistic is

$$T_{A_0} = \frac{{}_b\hat{\text{Acc}}_{O_\Lambda}^X - {}_b\text{Acc}_0}{\sqrt{(\frac{1}{M} + \frac{\rho}{1-\rho})S_{\text{Acc}}^2}}. \quad (3.31)$$

If $T_{A_0} \geq z_{1-\alpha}^*$, one can reject the null hypothesis H_0 and conclude that H_1 is true, i.e., that ${}_b\text{Acc}_{O_\Lambda}^X > {}_b\text{Acc}_0$.

Are the average accuracies of two selection algorithms $O_{\Lambda_1}^1$ and $O_{\Lambda_2}^2$ equal ? In this case, the null hypothesis H_0 is ${}_b\text{Acc}_1^X = {}_b\text{Acc}_2^X$ and the alternative H_1 is ${}_b\text{Acc}_1^X \neq {}_b\text{Acc}_2^X$. As for the stability, one could assume independence between the accuracy estimates of both selection methods and derive a statistic similar to Equation 3.25. However, the paired t-test methodology is much more powerful (but requires both methods to be evaluated on identical train and test samples). Instead of measuring the accuracy of a particular method in each selection run, we assess the accuracy difference between both algorithms and perform the hypothesis test on this difference, which also follows a Student's t distribution [NB99]. The test statistic is then

$$T_{A_{21}} = \frac{{}_b\hat{\text{Acc}}_2^X - {}_b\hat{\text{Acc}}_1^X}{\sqrt{(\frac{1}{M} + \frac{\rho_{21}}{1-\rho_{21}})S_{\text{Acc}_{21}}^2}}, \quad (3.32)$$

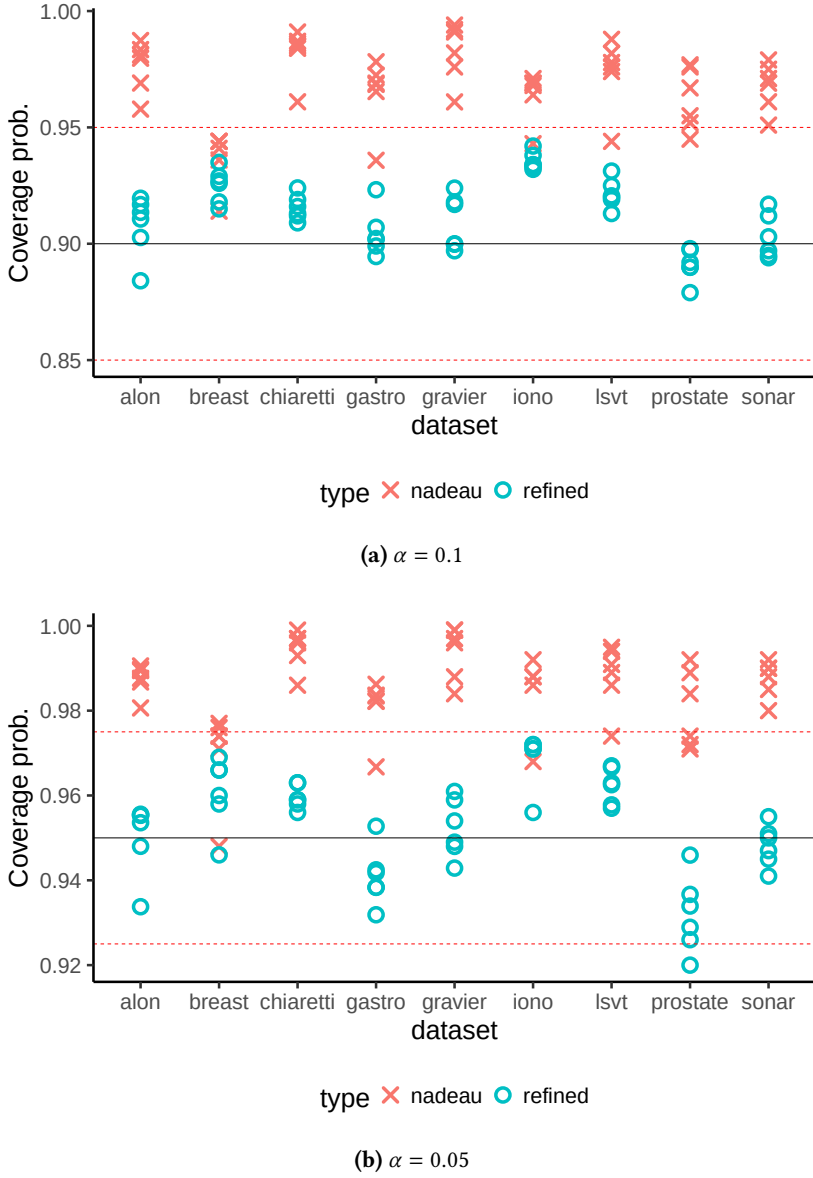


Figure 3.4: Coverage probabilities of the original confidence intervals of Nadeau and Bengio [NB99] and our refined methodology, applied to bound $b\text{Acc}_{O_{\lambda}}^X$. This is evaluated on 9 simulated data sets generated from true data with $M \in \{5, 10, 15, 30, 50, 100\}$. The number of repetitions is set to 1000. The confidence intervals of Nadeau and Bengio are overly conservative most of the time, while our proposal is much more satisfactory.

with ρ_{21} the correlation between the difference estimates and $S_{\text{Acc}_{21}}^2$ their estimated variance. Once again, the standard t-test puts ρ_{21} to 0 which leads to liberal inference, Nadeau and Bengio set it to $\rho_0 = \frac{n_2}{n}$ and we set it to $\rho_0 \hat{\rho}_s$ with $\hat{\rho}_s$ the estimated correlation between the difference in accuracy of both algorithms in different selection runs when evaluated on the shared examples. Instead of estimating the correlation between the random variables C_i^m of Equation 3.29, we define $\Delta C_i^m = C_i^m(O_{\Lambda_2}^2) - C_i^m(O_{\Lambda_1}^1)$ and estimate $\hat{\rho}_s = \rho(\Delta C_i^m, \Delta C_i^{m'})$ with $m \neq m'$ and i the index of an out-of-bag example in both runs, using the previously described bootstrap methodology. Similar to Figure 3.4, Figure 3.5 shows the coverage probabilities of the confidence intervals on the mean difference associated to Equation 3.32. The two compared selection algorithms are the LASSO on the one hand and the Elastic-Net penalty with several λ_2 values on the other hand. Accuracy differences range from below 1% to more than 10%. The corrected t-test is once again much too conservative for our purpose, while our approach is much more satisfactory. The estimator $\hat{\rho}_s$ mostly lies between 0.1 and 0.4. The applied correction $\frac{\rho}{1-\rho}$ thus ranges from ~ 3.5 to ~ 15 times less than the correction of Nadeau and Bengio.

3.4.3 Statistic tools of the expected satisfaction

Let now $\mathcal{P}$ be the multivariate random variable with the balanced accuracy and stability as dimensions. From the previous sections,

$$\Delta \hat{\Phi} \triangleq \frac{\hat{\Phi} - \Phi}{\sqrt{v(\hat{\Phi})}} \sim \mathcal{N}(0, 1) \text{ and } \Delta \hat{A} \triangleq \frac{b\hat{\text{Acc}}_{O_\Lambda}^X - b\text{Acc}_{O_\Lambda}^X}{\sqrt{(\frac{1}{M} + \frac{\rho}{1-\rho})S_{\text{Acc}}^2}} \sim t_{M-1}(0, 1). \quad (3.33)$$

To derive statistical tools on $\mathcal{P}$, we assume independence between $\Delta \hat{\Phi}$ and $\Delta \hat{A}$. The probability of observing $\Delta \hat{\mathcal{P}} = (\Delta \hat{A}, \Delta \hat{\Phi})$ can then be found by taking the product of the probability of observing each component, which can be derived from their respective distribution in Equation 3.33. This independence assumption is reasonable as $\Delta \hat{A}$ is mostly affected by the unknown test examples, which have no impact on $\Delta \hat{\Phi}$.

The definition of the hidden satisfaction Our definition of the hidden satisfaction S_a^h is inspired by the interactions with domain experts that occur in practice. Let $\hat{P}a_M^X$ be a Pareto front obtained on a data set X with M runs of feature selection. The true Pareto front Pa remains unknown. Let a domain expert with the satisfaction function S_a with parameter γ . A data scientist presents his results and the domain expert chooses the point $(\hat{a}_b, \hat{\Phi})$, from the estimated Pareto front $\hat{P}a_M^X$, which maximizes his satisfaction S_a . We define

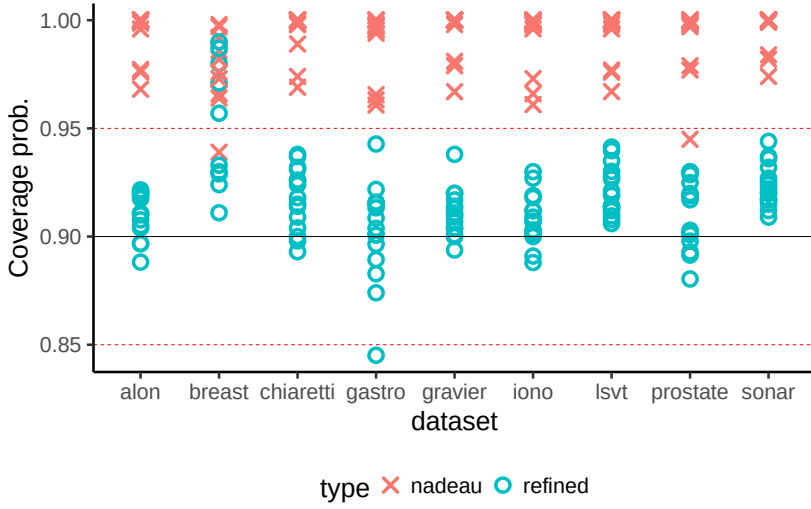
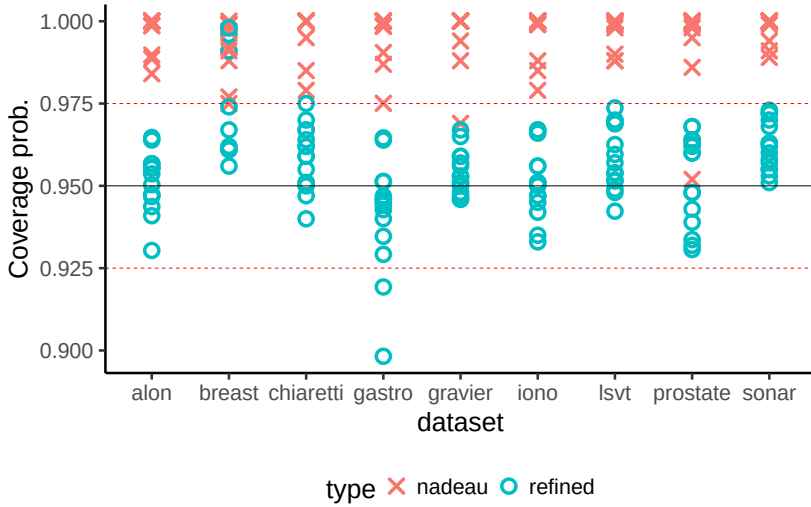
(a) $\alpha = 0.1$ (b) $\alpha = 0.05$

Figure 3.5: Coverage probabilities of the original confidence intervals of Nadeau and Bengio [NB99] and our refined methodology, applied to bound ${}_b\text{Acc}_{O_2}^X - {}_b\text{Acc}_{O_1}^X$ with O^1 the LASSO and O^2 the Elastic-Net logistic regression with several λ_2 values. This is evaluated on 9 simulated data sets generated from true data with $M \in \{5, 10, 15, 30, 50, 100\}$. The confidence intervals of Nadeau and Bengio are overly conservative (a significant fraction of their coverage probability is even equal to 1). The number of repetitions is set to 1000. Our proposal is much more satisfactory with only a few points overly conservative or liberal.

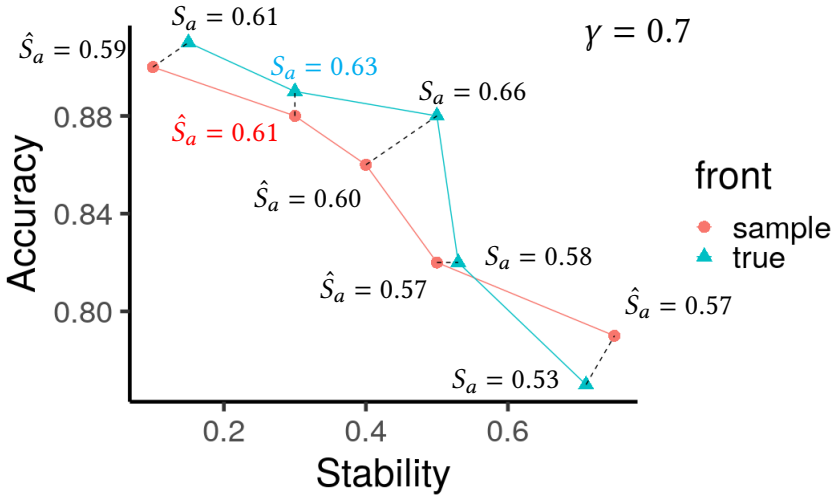


Figure 3.6: Illustration of the notion of hidden satisfaction. In this example, the domain expert has the satisfaction function S_a with parameter $\gamma = 0.7$. The sample Pareto front $\hat{P}a_M^X$ is in red. The domain expert picks the point that maximizes his satisfaction function, which is here the second point with the highest accuracy. His sample satisfaction $\hat{S}_a$ is equal to 0.61. The blue Pareto-front is the unknown true front. In this case, the true test accuracy has been underestimated ($\Delta A < 0$), and the hidden satisfaction of the expert is equal to 0.63. His optimal satisfaction would have been 0.66 had he made his choice on the true Pareto front.

the hidden satisfaction of this expert as his satisfaction, were he to know the true accuracy and stability of his chosen point O_Λ :

$$S_a^h = S_a(b\text{Acc}_{O_\Lambda}^X, \Phi_{O_\Lambda}^X). \quad (3.34)$$

Another choice *could be* to define the true satisfaction of the domain expert as his satisfaction had he made his choice on the true Pareto front Pa directly. However, in practical scenarios, the true Pareto front remains unknown when the choice is made. That definition would not assess the real and practical satisfaction of the expert. Instead, it would measure its optimal satisfaction had his choice been made with all information. This optimal satisfaction is of little interest in practice as it cannot be reached. It rather acts as an upper bound to the expert's hidden satisfaction. Figure 3.6 illustrates this notion of hidden satisfaction further.

To compute confidence intervals on S_a^h , one can use Monte-Carlo simulations. Concretely, N samples are drawn independently from both distributions in Equation 3.33 and the satisfaction S_a is computed for each pair. The lower (upper) bound of the $(1 - \alpha)$ confidence intervals is then the $\frac{\alpha}{2}$ ($1 - \frac{\alpha}{2}$) quantiles of the satisfaction values. Figure 3.7 validates our approach.

Is the hidden satisfaction S_a^h greater than a given value S_0 ? Monte-Carlo simulations are used to estimate the p-value: we draw N samples, compute S_a for each pair and find α such that $S_a \leq S_0$ for a fraction $1 - \alpha$ of the samplings.

Are the hidden satisfaction of two selection algorithms equal ? By simple development,

$$S_{a2}^h - S_{a1}^h = \gamma(b\text{Acc}_2^X - b\text{Acc}_1^X) + (1 - \gamma)(\Phi_2\sqrt{b\text{Acc}_2^X} - \Phi_1\sqrt{b\text{Acc}_1^X}). \quad (3.35)$$

As previously, Φ_1 and Φ_2 are assumed independent to each other and to the accuracy measurements. However, $b\text{Acc}_2^X$ and $b\text{Acc}_1^X$ can be strongly correlated. By assuming that $t_{M-1}(0, 1) \approx \mathcal{N}(0, 1)$, we can find $\rho_{a1,a2} = \frac{\sigma_{a2}^2 + \sigma_{a1}^2 - \sigma_{a2-a1}^2}{2\sigma_{a1}\sigma_{a2}}$ and sample $b\text{Acc}_2^X$ and $b\text{Acc}_1^X$ from a bivariate normal distribution. Once again, the Monte-Carlo methodology is used to estimate the p-value of the hypothesis test. Figure 3.8 studies the power of the confidence intervals of the difference between hidden satisfactions. There is a tendency to conservatism that is probably due to the conservatism of the confidence intervals on the difference of stability proposed by Nogueira et. al. [NSB17a].

Note that when accounting for randomness, the optimal decision function of a domain expert (Equation 3.7) changes slightly and becomes

$$(\tilde{a}_b^*, \Phi^*) = \underset{(\hat{a}_b, \hat{\Phi})}{\operatorname{argmax}} E_{\Delta\hat{\Phi}}[S_a^h(\tilde{a}_b, \Phi)]. \quad (3.36)$$

The preferred trade-off is the one that maximizes the expected value of the hidden satisfaction. This is particularly important when the satisfaction function is not continuous. It would be very risky for a domain expert with the satisfaction function of Equation 3.12 to choose a trade-off with $\hat{a}_b \rightarrow \gamma$, as even a $\Delta\hat{A}$ slightly higher than 0 would result in a null true satisfaction. Let $\gamma(\hat{\mathcal{P}})$ be the set of γ values for which $\hat{\mathcal{P}}$ maximizes this expectation of the satisfaction function S_a (Equation 3.36). The expected hidden satisfaction of the Pareto-front Pa is then

$$E_{S_a^h}(Pa) = \sum_{\mathcal{P}=(\tilde{a}_b, \Phi)} \int_{\gamma(\hat{\mathcal{P}})} S_a^h(\tilde{a}_b, \Phi, \gamma) d\gamma. \quad (3.37)$$

This is illustrated in Figure 3.9. Note that under this formulation, the satisfaction functions 3.12 and 3.14 do no longer allow to measure the hypervolume. Yet, as $M \rightarrow \infty$, the variance of $\Delta\Phi$ tends to 0, and the satisfaction function 3.14 can still be used to measure the hypervolume and bound its true value under our proposed framework. For this reason, satisfaction function 3.14 is used in this document when performing statistical analysis on the hypervolume.

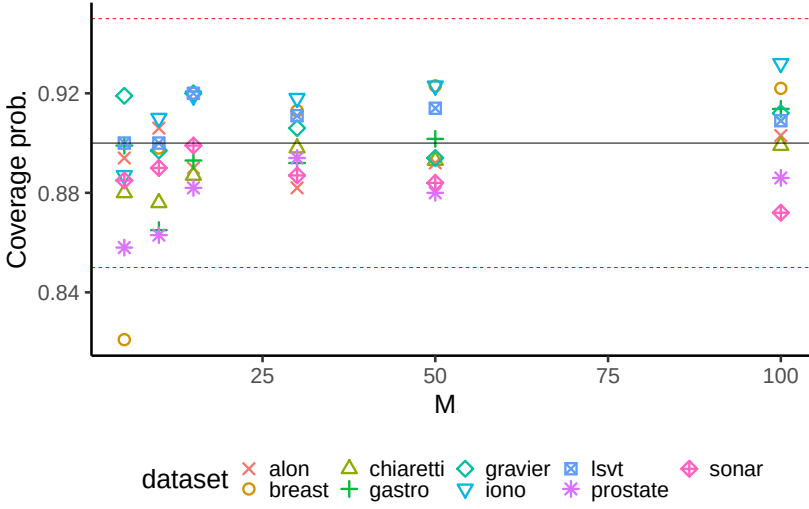
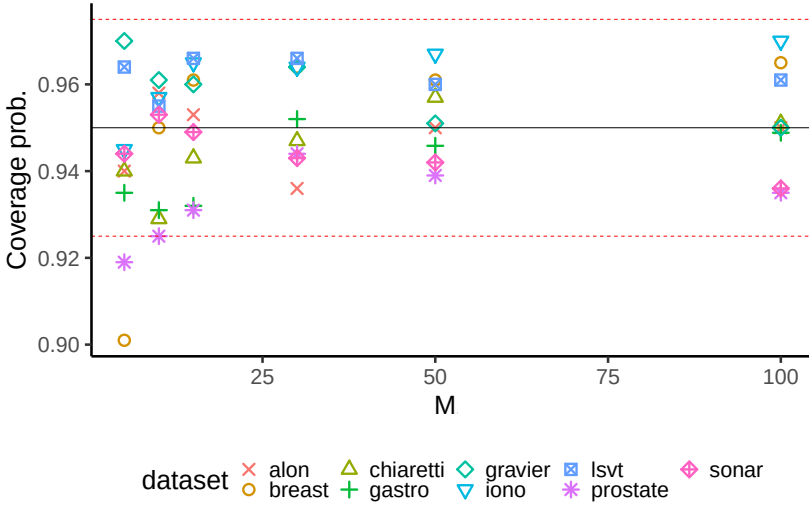
(a) $\alpha = 0.1$ (b) $\alpha = 0.05$

Figure 3.7: Coverage probabilities of our proposed confidence interval on S_a^h . The selection algorithm is the LASSO. This is evaluated on 9 simulated data sets generated from true data with $M \in \{5, 10, 15, 30, 50, 100\}$.

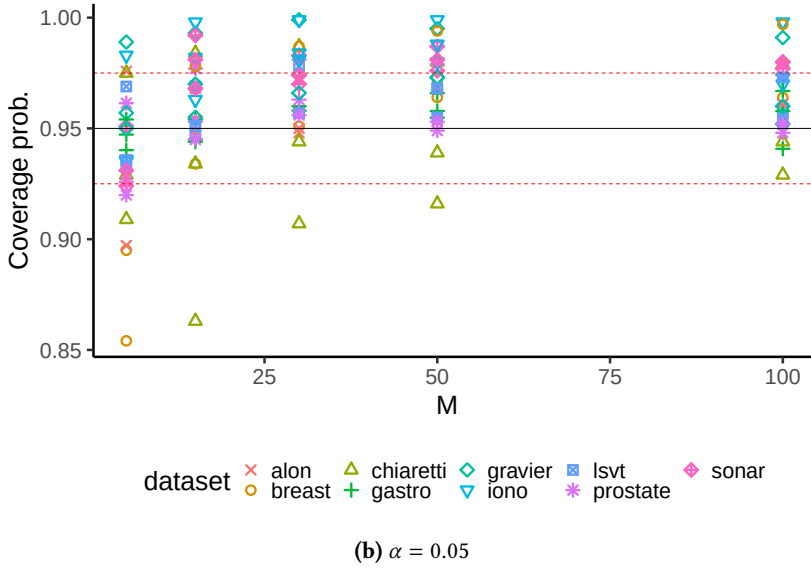
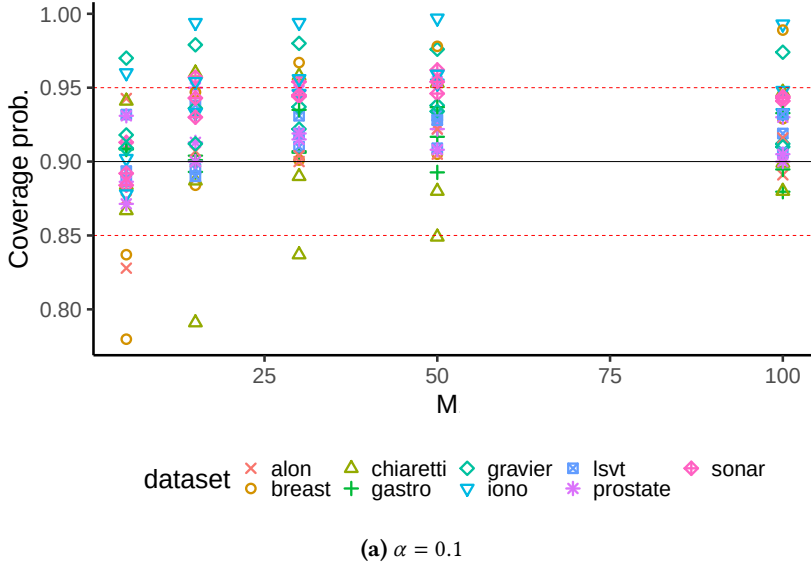


Figure 3.8: Coverage probabilities of our proposed confidence interval on $S_{a2}^h - S_{a1}^h$. The selection algorithms are the LASSO (1) and the Elastic-Net logistic regression with several λ_2 values (2). This is evaluated on 9 simulated data sets generated from true data with $M \in \{5, 15, 30, 50, 100\}$.

Inference about $E_{S_a^h}$ can be performed using Monte-Carlo simulations to bound each integral in Equation 3.37. The bounds on $E_{S_a^h}$ are then found by simple summation. Figure 3.10 validates the confidence intervals on $E_{S_a^h}$ for the Elastic-Net logistic regression. The Pareto-curve is drawn by varying the λ_2 parameter (see Section 3.5 for a concrete example of an Elastic-Net Pareto-front). Note that because the confidence interval on $E_{S_a^h}$ is computed through summation of the bounds of each Pareto-point, it could become overly conservative. Indeed, we here essentially assume the worst-case scenario where all Pareto points fall off their confidence intervals jointly. This conservatism should heavily depend on the evaluated Pareto-front and, more specifically, on the correlation between the Pareto points. Such a phenomenon is not largely present for the Elastic-Net regression. Comparison of Figure 3.7 and Figure 3.10 still shows that inference on E_{S_a} is slightly more conservative than inference on S_a .

The difference between the expected hidden satisfactions of two Pareto fronts can be written

$$E_{S_a^h}(Pa_2) - E_{S_a^h}(Pa_1) = \sum_{\substack{\mathcal{P}_1=(\tilde{a}_{b1}, \Phi_1), \\ \mathcal{P}_2=(\tilde{a}_{b2}, \Phi_2)}} \int_{\gamma(\hat{\mathcal{P}}_1, \hat{\mathcal{P}}_2)} S_a^h(\tilde{a}_{b2}, \Phi_2, \gamma) - S_a^h(\tilde{a}_{b1}, \Phi_1, \gamma) d\gamma, \quad (3.38)$$

with $\gamma(\hat{\mathcal{P}}_1, \hat{\mathcal{P}}_2)$ the set of γ values for which $\hat{\mathcal{P}}_1$ maximizes Equation 3.36 on the Pareto front Pa_1 and $\hat{\mathcal{P}}_2$ maximizes Equation 3.36 on Pa_2 . For inference about $E_{S_a^h}(Pa_2) - E_{S_a^h}(Pa_1)$, the bounds on each term of the sum are found with Monte-Carlo simulations and are summed.

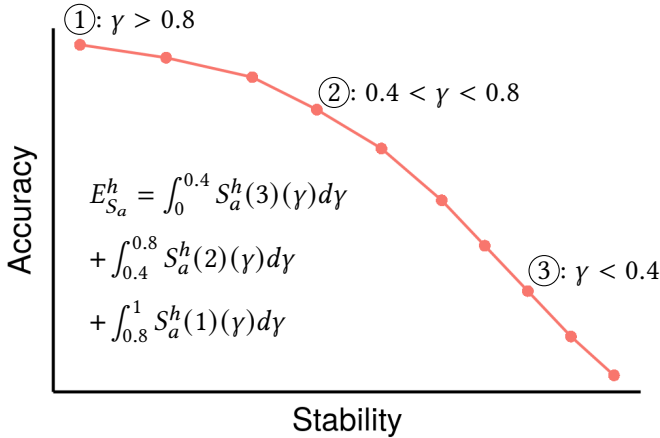


Figure 3.9: Toy illustration of the expected hidden satisfaction of a Pareto-front. The first point is preferred by experts with $\gamma > 0.8$, the second point by experts with $0.4 < \gamma < 0.8$, and the third point by experts with $\gamma < 0.4$. The expected hidden satisfaction is the sum of three terms, each term being the integral of the hidden satisfaction of a specific point over its γ range.

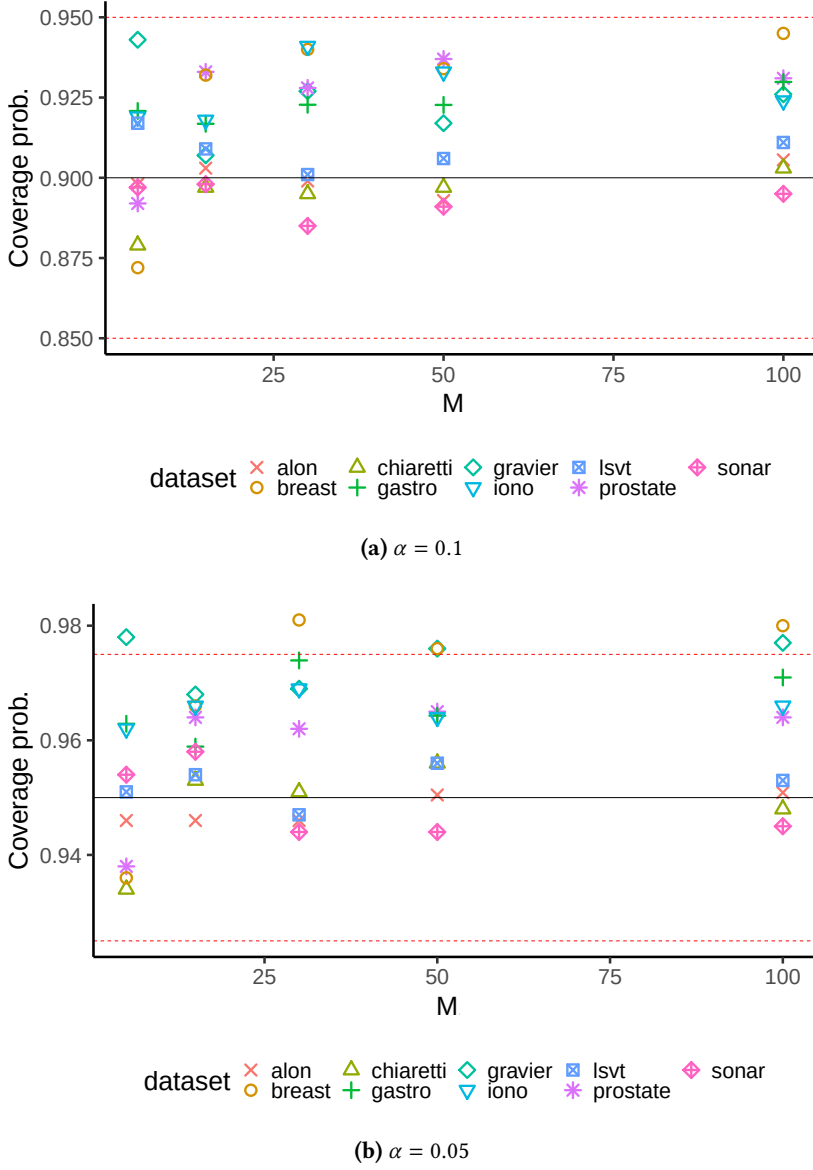


Figure 3.10: Coverage probabilities of our proposed confidence interval on E_{S_a} . The selection algorithm is the Elastic-Net logistic regression with several λ_2 values. This is evaluated on 9 simulated data sets generated from true data with $M \in \{5, 15, 30, 50, 100\}$.

3.5 A concrete example

A simple way to explore the bi-objective space is to perform *model selection* on standard machine learning techniques. Different meta-parameter choices might lead to different $(\tilde{a}_b, \Phi)$ compromises, from which a Pareto-optimal subset can be extracted. Nonetheless, there currently exists no ML approach with a parameter *explicitly designed* to balance between accuracy and stability.

Figure 3.11 illustrates this model selection on the `alon` microarray data set (introduced later) with the elastic-net. The elastic-net, introduced in Section 2.2.2.2, has two meta-parameters: λ_1 , which dictates the overall regularization intensity, and λ_2 , which balances between the $L1$ and $L2$ penalties. We here make λ_2 vary by steps of 0.05 and set λ_1 such that the number of selected features is, on average over the $M = 100$ runs, equal to 20. The four first points are obtained with $\lambda_2 = 1$ (LASSO), $\lambda_2 = 0.95$, $\lambda_2 = 0.9$ and $\lambda_2 = 0.85$ and are not Pareto-optimal. The optimal accuracy is reached with $\lambda_2 = 0.8$. Increasing λ_2 further decreases the accuracy, but stability continues to increase, which draws the Pareto-optimal curve. As can be seen, the accuracy plummets quite fast when stability increases and no large stability values can be reached. The hypervolume is very poor (0.215 ± 0.08 with significance level $\alpha = 0.05$) while the expected satisfaction is better but still disappointing (0.41 ± 0.115). Chapter 4 introduces a new algorithm with better performance. Note that despite the relatively high number of selection runs $M = 100$, the uncertainty on E_{S_a} and HV remains high as this data set has only $n = 62$ examples. The accuracy can thus not be estimated precisely. Note that, unlike Figure 3.1, Figure 3.11 does not display the complete objective space for visualisation reasons.

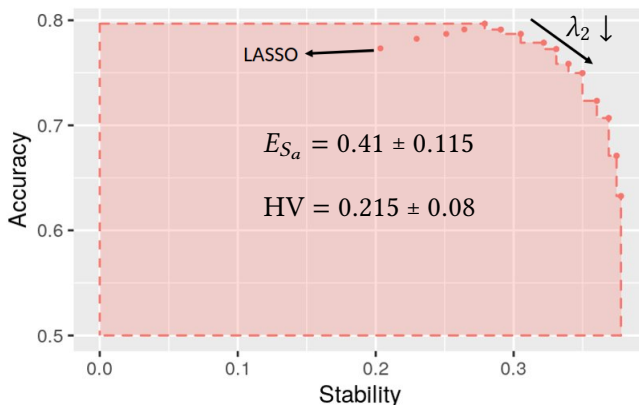


Figure 3.11: Performance evaluation of the elastic-net on the `alon` data set. The α meta-parameter allows some form of (accuracy, stability) control, but to a very limited extent.

The logistic-RFE is another selection algorithm introduced in Sec-

tion 2.2.2.4. On a long run, it produces a single Pareto-point: ($\hat{a}_b = 0.828$, $\hat{\Phi} = 0.40$). Its expected satisfaction is better than the one of the Elastic-Net ($E_{S_a} = 0.492 \pm 0.12$). Even though both confidence intervals are largely overlapping, our paired test methodology allows us to conclude that the expected satisfactions are different with a p-value below $1e - 3$.

3.6 Discussion

This chapter proposes an evaluation framework that estimates the overall quality of a feature selection algorithm. The objectives of prediction performance and selection stability are aggregated into a single measure (noted S_a), designed to represent the satisfaction of domain experts with respect to a specific (accuracy, stability) trade-off. The expected satisfaction metric of a complete Pareto-optimal front (noted E_{S_a}) is then defined as the average satisfaction of domain experts regarding their preferred trade-off from the evaluated front.

The proposed bi-objective framework is generic and can be applied to any class of satisfaction function S_a . For instance, the well-known hypervolume indicator can be re-derived in this framework with a specific choice of satisfaction function. In this document, we assume a particular form for these satisfaction functions based on four axioms (one of which the hypervolume violates). Yet, a multitude of functions satisfy these axioms and are thus valid candidates as well. Furthermore, the subjective preferences of domain experts for both objectives are modeled through a parameter γ whose probability density is assumed to be uniform, without any other knowledge. It is unlikely to be the case in practice. A possible refinement of our framework could be made through a survey on actual domain experts. They would be asked to estimate their γ parameter based on the contour plots of the resulting satisfaction function (see Figure 3.2). Such a survey would allow a more accurate estimation of the true probability density function of γ .

As predictive accuracy and selection stability are measured on finite data and using a finite number of selection runs M , the reported values are only estimates that come with an uncertainty. Existing inference mechanisms for both objectives are reviewed and refined for the accuracy. More specifically, the strong assumption of Nadeau and Bengio that the correlation ρ between the M out-of-bag accuracies is equal to the expected fraction of shared test examples $\frac{n_2}{n}$ is relaxed. The correlation ρ is estimated on the data instead. This can be done when one aims to bound the expected accuracy of models learned on samples of a specific data set. Our methodology has been shown to be more powerful than the original work of Nadeau and Bengio for this purpose.

From these single-objective inference mechanisms, we derive statistical

tools for the hidden satisfaction of domain experts (S_a^h) and its expected value ($E_{S_a^h}$). Such tools include confidence intervals and hypothesis tests that quantify the statistical significance of observed differences between Pareto-fronts. These tools are successfully validated on pseudo-real data with various sample-to-feature ratios and class imbalances. They should help data scientists compare different feature selection algorithms, favoring the ones with higher average satisfaction.

An observation that could be seen as a limitation is that inference is performed on the hidden satisfaction of an algorithm when this algorithm is trained on a fixed data set X , not on any data set X' of the same size as X and issued from the same generative distribution. In other words, inference is made about the accuracy of the predictive models and about the stability of the selected features on which domain analysis is performed. This can be argued as the correct strategy. The validity of the domain analysis should be much more impacted by the accuracy and stability of the studied models and features than by the accuracy and stability obtained when the same algorithm is trained on another hypothetical data set from the same distribution. Whenever predictive models learned on X are accurate and stable, the selected features are likely to represent interesting biomedical functions and should be worth investigating.⁴ This is true even if the same algorithm fails to learn accurate models on X' . Yet, the reader must be aware of this subtlety as it impacts the scope of the statistical analyses that are performed and the statistical significance levels that are reported. There currently exists no statistical tools for inference about the average stability of a learning algorithm when trained on different data sets issued from the same distribution. One could naturally use a nested bootstrap protocol to derive multiple stability values. However, these stability values would be estimated with overlapping train sets, and the derivation of the corrections needed to account for these overlaps remains an open challenge.

Finally, we must note that the proposed evaluation framework suffers from a small optimistic bias. This comes from the nature of choice between different trade-offs and arises as soon as more than one set of meta-parameter is evaluated. Points that have been (by chance) overestimated have a higher chance of maximizing the satisfaction function of domain experts than points that have been underestimated (even if all points are Pareto-optimal). The overall expected satisfaction is thus naturally slightly upwardly biased. For this reason, in this document, the number of evaluated points is usually kept small and similar between the compared approaches. When this can not be ensured (most notably in Chapter 4), we use very high number of selection runs (e.g., $M = 1000$), which already reduces the bias considerably. Note that

⁴Note that the term accurate refers here to the accuracy on the whole population, and not to its estimated value on the out-of-bag samples in each run.

the smoothness of the curve in Figure 3.11 indicates that this bias is certainly tiny for the elastic-net when evaluated on `alon` with $M = 100$. We let the design of an unbiased protocol, for instance with some kind of nested validation or unique validation/test split, as future work. One must be mindful that the satisfaction function used in this document is a non-linear function of the accuracy. It can thus not be expressed as the average of several satisfactions evaluated on, for instance, multiple runs of a nested protocol.⁵ A unique validation/test split protocol would substantially increase the already large variance of the estimates in the *small-n-big-d* setting. It would also potentially create a negative bias as the choices of domain experts would be made on fewer data points.

⁵Indeed, $E[S_a(\tilde{a}_b, \Phi)] \neq S_a(E[\tilde{a}_b], \Phi)$ as S_a is non linear with respect to $\tilde{a}_b$.

The hybrid-RFE algorithm

4

This chapter proposes the hybrid-RFE algorithm, which draws a Pareto-optimal front in the (accuracy, stability) bi-objective space. Domain experts can choose a particular compromise from this Pareto front based on their personal preferences. The fronts are obtained by pre-selecting some features based on a stable univariate criterion before running the multivariate recursive feature elimination (RFE) algorithm, which selects the most appropriate additional features. Section 4.1 formally defines the hybrid-RFE algorithm. Section 4.2 evaluates the algorithm on 6 microarray and one mass-spectrometric data sets. Results show that large stability increases are obtained at a small cost of classification accuracy. Finally, Section 4.3 studies alternative bias strategies and provides some theoretical justification as to why *forcing* the selection of the stable features is the optimal strategy. Most of this chapter has been published in [HD20].

4.1 The Algorithm

The methodology is summarized in Algorithm 3. It is based on the classic Recursive Feature Elimination algorithm (Algorithm 2, presented in Section 2.2.2.4). First, a set of *stable features*, S_N , is found as the top- N features based on a univariate criterion (lines 3,4). Univariate filters tend to be more stable than multivariate methods as they do not consider feature interdependencies. These N features are then forced to be selected at each iteration of the RFE, which selects, in a multivariate fashion, the most appropriate additional features. It does so by iteratively minimizing the logistic loss (line 7), ranking every feature (but the ones of the stable set) based on the absolute value of their weight $\mathbf{w}$ in the learned decision function (line 8), and dropping the one feature with minimal weight (line 9), until the desired number of features k is reached.¹ Finally, it learns the final decision function by minimizing the logistic loss on the k selected features (line 10), possibly with a different regularization constant λ_f . The main difference between Algorithm 3 and

¹For computational reasons, it is common to drop a fraction of the remaining features instead of a single one at each iteration. We opt here to drop 20% of the remaining features at each pruning step.

the classic RFE algorithm is that the features in S_N are never dropped and are thus always present in the final model. To take advantage of this knowledge, one can apply differential shrinkage to these features to increase their importance in the multivariate selection (line 7, with $\odot$ the element-wise product). The intensity of this differential shrinkage is dictated by the meta-parameter $\epsilon \leq 1$ used in line 5.

Whenever the set of *stable features* S_N is robust, increasing its size N , i.e., the number of features selected beforehand, is expected to increase the overall selection stability at the cost of a possible decrease in predictive accuracy. If $N = 0$, this hybrid-RFE is equivalent to the classical RFE, for which no feature is pre-selected (except that the logistic loss is considered here instead of the often used hinge loss). This logistic loss choice is motivated by the stability gains it tends to offer, as studied in Appendix C. When $N = k$, the approach becomes equivalent to a purely univariate filter. The Pareto curve is drawn by increasing N from 0 to the number of selected features k .

Algorithm 3 Hybrid RFE.

```

1: procedure SELECTFEATURES( $N, \lambda, \epsilon, \lambda_f$ )
2:    $\mathcal{F} \leftarrow$  the set of all features
3:    $r_f \leftarrow$  univariate criterion rank of each feature (descending order)
4:    $S_N \leftarrow \{f : r_f \leq N\}$ 
5:    $\beta_f \leftarrow \epsilon$  if  $f \in S_N$ , 1 otherwise
6:   while  $|\mathcal{F}| > k$  do
7:      $\mathbf{w}^* \leftarrow \operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^n \log(1 + \exp^{-y_i(\mathbf{w}\mathbf{x}_i)}) + \lambda \|\boldsymbol{\beta} \odot \mathbf{w}\|_2$ 
8:      $\mathbf{r}^* \leftarrow$  rank features  $\{f \in \mathcal{F} \setminus S_N\}$  on  $|\mathbf{w}_f^*|$  in descending order
9:      $\mathcal{F} \leftarrow \mathcal{F} \setminus \{f : r_f^* = |\mathcal{F}| - N\}$ 
10:   $\mathbf{w}^* \leftarrow \operatorname{argmin}_{\mathbf{w}} \sum_{i=1}^n \log(1 + \exp^{-y_i(\mathbf{w}\mathbf{x}_i)}) + \lambda_f \|\mathbf{w}\|_2$ 
11:  return  $(\mathcal{F}, \mathbf{w}^*)$ 

```

4.2 Evaluation on microarray data

Our experiments are focused on two univariate criteria: the supervised Welch's t-test [Wel47] and the unsupervised sample variance. The Welch's T-test statistic quantifies the significance level of the observed difference between the means of the two classes (see Section 2.2.1.2). We observed experimentally that, unlike the variance, which is very stable, the Welch's t-test is generally only slightly more stable than the multivariate RFE selection, making any compromise between the two difficult. To generate a family of univariate filters with different (relevance, stability) trade-offs, we pose the

name	year	n	n_{maj}	d	disease/cancer	d post-fl.
alon	1999	62	40	2000	colon	2000
borovecki	2005	31	17	22283	Huntington's	1000
singh	2002	102	52	12600	prostate	1250
gravier	2010	168	111	2905	breast	2905
chiaretti	2004	128	74	12625	leukemia	5000
chin	2006	118	75	22215	breast	100
arcene	2003	198	111	10000	ovarian/prostate	5000

Table 4.1: Information on used data sets, from the UCI machine learning repository (arcene) and from the datamicroarray R package for the others. The number of samples of the majority class is noted n_{maj} .

following criterion

$$TV_f = \frac{|\mu_+(f) - \mu_-(f)|}{\sqrt{\frac{\sigma_+^2(f)}{N_+} + \frac{\sigma_-^2(f)}{N_-}}} + \lambda_v \text{var}(f), \quad (4.1)$$

with λ_v a parameter balancing between the weight given to the Welch's t-test and the sample variance. A low λ_v generates a relevant but not very stable univariate filter, while a high λ_v defines a less relevant but more stable criterion. Experiments are performed on six microarray and one mass-spectrometric data sets, summarized in Table 4.1. All these data sets have a small n (number of samples) to d (number of features) ratio, which generally causes feature selection methods to be particularly unstable. The learning task consists of predicting whether a patient is suffering from the corresponding disease. As is often done when dealing with high dimensional data sets, we first pre-filter the feature space by removing the features with the lowest variance (except for alon and gravier, for which such a pre-filtering has already been performed). The amount of pre-filtering is found such as to maximize the predictive performance of the classical RFE ($N = 0$) and is kept constant for all values of N , λ , ϵ , and λ_f .

Results with $k = 20$ and $M = 1000$ can be seen on Figure 4.1. The plot represents the areas dominated by the Pareto-optimal curves that are drawn by model selection on λ and λ_f of the classic RFE (purple), the hybrid variance RFE ($\lambda_v \rightarrow \infty$) (red) and the hybrid RFE with two other values of λ_v (cyan and green). The hybrid-RFE is able to increase the selection stability by considerable amounts, sometimes even without decreasing the predictive accuracy at all. Our approach strictly dominates model selection for all data sets. Apart from some small performance increases for low stabilities, better compromises are reached when $\lambda_v \rightarrow \infty$, making our approach mostly sensitive to the stability of the *stable set*, and less to its predictive accuracy.

On borovecki and chin, the best Pareto curve is obtained by increasing λ_0 gradually and fixing $N=k$. In other words, there appears to be no benefit to using multivariate selection on these two data sets.

Figure 4.2 shows that the hybrid-RFE is able to select features that are complementary to the ones whose selection is forced. The blue area is obtained by selecting the remaining features independently of the pre-selected ones. The green (resp. red) area is obtained with the hybrid-RFE without (resp. with) differential shrinkage. Our approach dominates the independent selection, and differential shrinkage can improve the compromise further by 1) helping the RFE select adequate complementary features, hence increasing the predictive accuracy, and 2) stabilizing the selection of the complementary features. This can be seen in Figure 4.2a where points inside a specific circle are obtained with the same value of N (here 12 and 15).

Table 4.2 compares the expected satisfaction and hypervolume values of the classic RFE and its hybrid extension. Both metrics are largely increased, especially the hypervolume, which is often doubled. Results are statistically significant. Table 4.3 studies the difference in performance between the hybrid-RFE and some baseline approach combining the Pareto-front drawn via the classic RFE and the two points obtained with the t-test and the full variance univariate criteria. This Table represents the added value of our approach with respect to this simple baseline. It shows that, once again, the hypervolume tends to be increased by larger amounts than the expected satisfaction. Yet, the latter is still sometimes largely increased (e.g., +10% on borovecki, +8% on gravier and singh). On other data sets, the increase is more marginal (e.g., +3% on alon, +2% on chiaraetti). As the hybrid-RFE draws a Pareto-optimal curve from the classic RFE to the full variance criterion, its advantage in performance over the baseline naturally tends to decrease when the variance has strong predictive accuracy. For instance, it nearly completely vanishes on chin because of this phenomenon (there is no real compromise to be made). Due to the small sample sizes (see Table 4.1), results are not always statistically significant. Results are similar for $k = 10$, although generally more statistically significant, as depicted in Table 4.4.

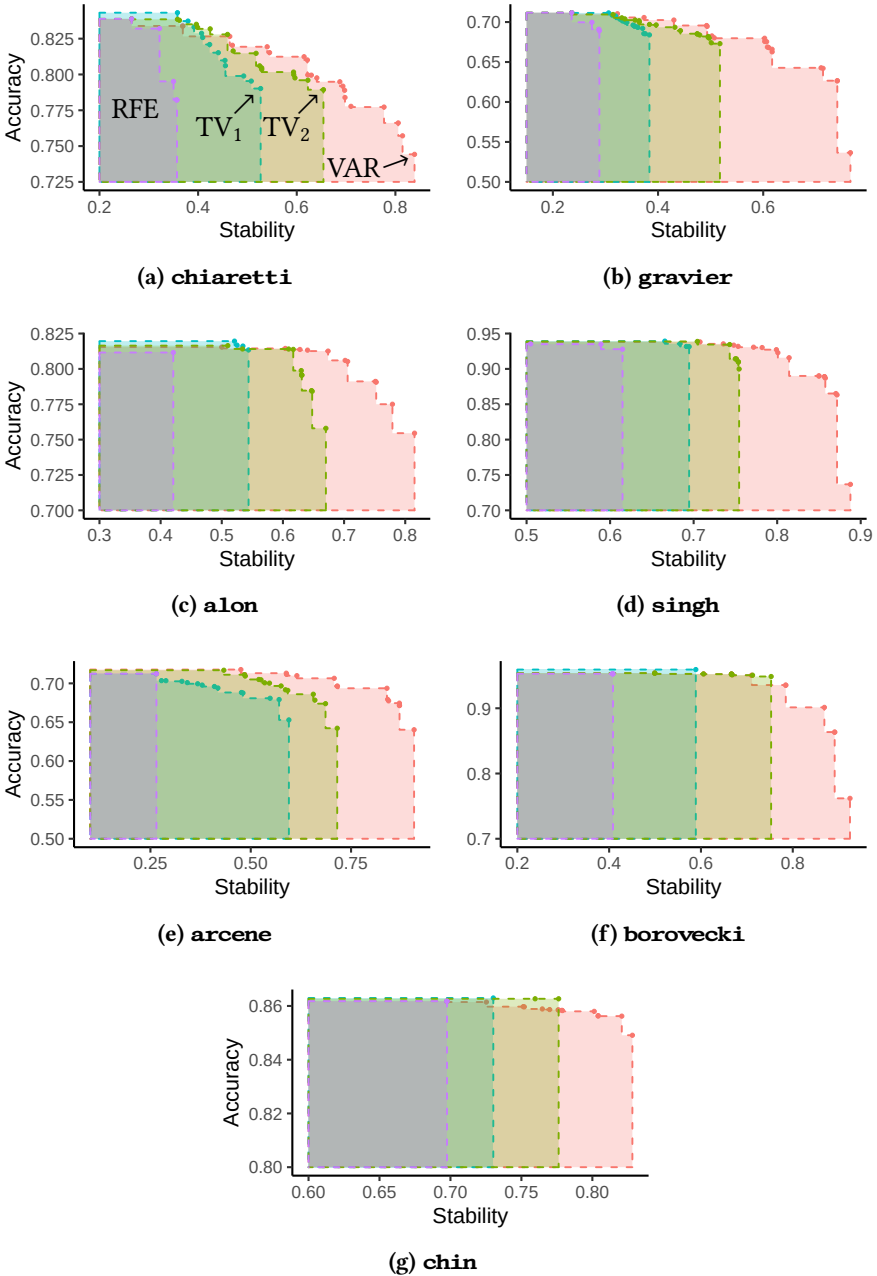


Figure 4.1: Evaluation of the classic RFE (purple), the variance-hybrid RFE (VAR) (red), and some ttest-variance-hybrid RFE (TV1,TV2) (cyan and green). For the sake of readability, the axes of the different plots are not at the same scale.

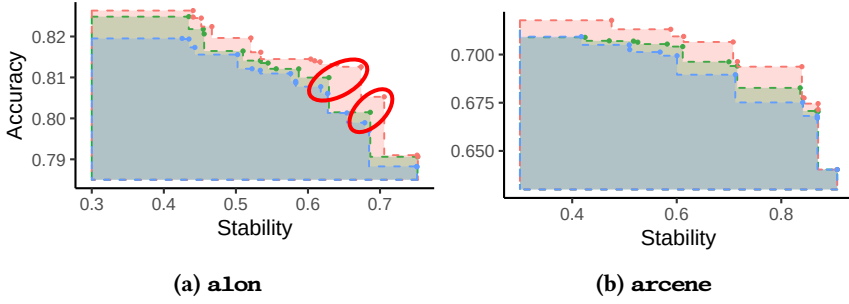


Figure 4.2: Comparison of the independent selection (blue), the hybrid RFE with no differential shrinkage (green) and with differential shrinkage (red). Points inside a given circle correspond to the same value of N (here 12 and 15).

	E_{S_a}			HV		
name	RFE	hRFE	pval	RFE	hRFE	pval
boro.	0.65±0.08	0.82±0.12	2.5e-3	0.37±0.06	0.82±0.14	<1e-3
gra.	0.29±0.06	0.37±0.07	<1e-3	0.12±0.03	0.29±0.08	<1e-3
alon	0.48±0.13	0.59±0.16	2.5e-3	0.27±0.10	0.51±0.18	<1e-3
arcene	0.30±0.06	0.46±0.08	<1e-3	0.11±0.03	0.37±0.09	<1e-3
singh	0.72±0.06	0.80±0.07	<1e-3	0.53±0.06	0.76±0.09	<1e-3
chia.	0.47±0.06	0.59±0.08	<1e-3	0.24±0.04	0.53±0.09	<1e-3
chin	0.66±0.10	0.70±0.11	<1e-3	0.50±0.10	0.60±0.12	<1e-3

Table 4.2: Performance comparison of the classic RFE and the hybrid-RFE algorithm.

	E_{S_a}			HV		
name	base	hRFE	pval	base	hRFE	pval
boro.	0.72±0.13	0.82±0.12	0.075	0.65±0.18	0.82±0.15	0.075
gra.	0.29±0.07	0.37±0.07	1e-3	0.15±0.07	0.29±0.08	<1e-3
alon	0.56±0.16	0.59±0.16	0.3	0.46±0.16	0.51±0.18	0.125
arcene	0.40±0.10	0.46±0.08	0.075	0.29±0.11	0.37±0.09	0.02
singh	0.72±0.07	0.80±0.07	<1e-3	0.66±0.10	0.76±0.09	5e-3
chia.	0.57±0.10	0.59±0.09	0.15	0.47±0.09	0.53±0.09	0.025
chin	0.70±0.10	0.70±0.11	0.275	0.59±0.12	0.60±0.12	0.225

Table 4.3: Performance comparison of the hybrid-RFE algorithm and the baseline obtained by building the RFE-ttest-variance Pareto-optimal front.

	E_{S_a}			HV		
name	base	hRFE	pval	base	hRFE	pval
boro.	0.65±0.14	0.81±0.13	7.5e-3	0.5±0.24	0.80±0.18	2.5e-3
gra.	0.27±0.06	0.32±0.07	1e-3	0.13±0.06	0.22±0.06	<1e-3
alon	0.51±0.15	0.54±0.16	0.3	0.40±0.14	0.43±0.15	0.15
arcene	0.36±0.09	0.44±0.09	<1e-3	0.24±0.10	0.36±0.10	<1e-3
singh	0.69±0.07	0.79±0.07	<1e-3	0.62±0.11	0.77±0.09	<1e-3
chia.	0.54±0.09	0.55±0.09	0.2	0.44±0.10	0.49±0.09	0.035
chin	0.70±0.10	0.70±0.11	0.275	0.59±0.12	0.60±0.12	0.225

Table 4.4: Performance comparison of the hybrid-RFE algorithm and the baseline obtained by building the RFE-ttest-variance Pareto-optimal front, for $k = 10$.

4.3 Alternative bias strategies

The hybrid-RFE *forces* the selection of the top- N features based on a univariate criterion. When $N = 0$, it is equivalent to the classical logistic-RFE, with strong accuracy but poor stability. When $N = 20$, it becomes equivalent to the univariate criterion, with lower accuracy but better stability. A Pareto curve is drawn by increasing N from 0 to the number of selected features k . In this section, we justify theoretically and experimentally why the choice of *forcing* the selection of these $N < k$ features is actually the *optimal* strategy.

We first experimentally compare this *forcing* approach to two other simple biasing schemes that draw Pareto curves starting and ending on the same points:

1. The *increasing filtering* approach prevents the selection of the N_2 bottom-ranked features according to the same univariate criterion. By gradually increasing N_2 from 0 ($\Leftrightarrow$ classical RFE) up to $d - k$ ($\Leftrightarrow$ univariate criterion), another Pareto-optimal curve is drawn.
2. The *fixed biased set* technique never forces the selection of any feature but rather increases the bias put on the top- k features. We perform that using the ϵ parameter of the hybrid-RFE. More specifically, we reduce it from $\epsilon = 1$ ($\Leftrightarrow$ classical RFE) to $\epsilon = 0$, which, for λ sufficiently high, actually forces the selection of these top- k features ($\Leftrightarrow$ univariate criterion).

We perform this experiment on the singh data set. To isolate the effect of N and N_2 , we pose $\lambda_o = \infty$ (variance criterion) and $\epsilon = 1$ (for the *forcing* strategy). Results are depicted on Figure 4.3.

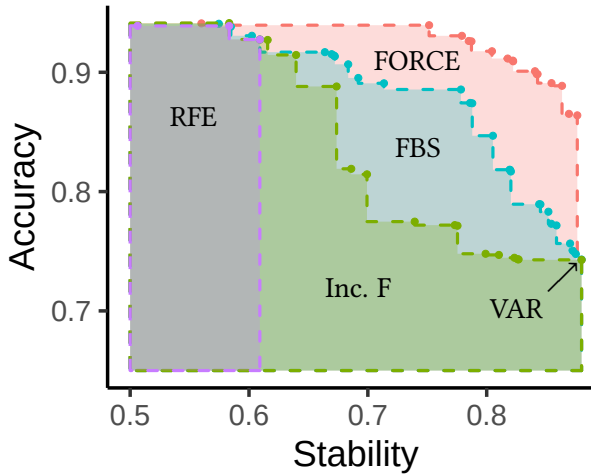


Figure 4.3: Comparison of the three studied biased schemes. The *increasing filtering* is the worst approach because it prevents the selection of potentially relevant features. The *forcing* strategy outperforms the *fixed biased set*, as justified by Theorem 1.

The Pareto-optimal curves of all three approaches all start and end on the same points. However, the path they follow varies greatly. The *increasing filtering* is clearly the worst way to balance between accuracy and stability. As N_2 is reduced more and more, the selection of some (possibly highly) relevant features is prevented. This can cause a strong accuracy decrease but also some stability reduction. There is no guarantee to increase stability at all until N_2 converges towards k . The other two approaches never prevent the selection of a feature. Figure 4.3 also shows that the *forcing* approach outperforms the *fixed biased set* one. This is theoretically justified below.

Theorem 1 formalizes that forcing the selection of the biased features in all selection runs is the optimal strategy in an idealized and simplistic scenario. First, let us assume that we have at our disposal a criterion that allows the identification of a subset of features, $\mathcal{F}_b$, with perfect stability: $\Phi_{\mathcal{F}_b} = 1$. In practice, the variance criterion has a stability between 0.8 and 0.9 on the microarray data sets under study. These features are identified reliably but are less informative. We assume that the marginal accuracy drop caused by their selection increases with their number. This assumption is actually quite well respected in practice, as can be seen in Figure 4.1. The selection of a few biased features causes only small marginal drops in accuracy, marginal drops that increase when the number of forced features increases. In that scenario, Theorem 1 states that the existence of two distinct features $f_1, f_2 \in \mathcal{F}_b$ with selection frequencies lower than 1, *i.e.*, whose selection is *not* forced, cannot lead to a Pareto-optimal compromise. Consequently, the Pareto-optimal

curve is obtained by gradually increasing the number of features from $\mathcal{F}_b$ that are forced to be selected in every run. In Theorem 1, the binary variable $z_{i,f}$ indicates the selection or non-selection of feature f in run i .

Theorem 1. *Let $\mathcal{F}_r$ be the set of features that are selected at least once by a relevant feature selection algorithm. Let $\mathcal{F}_b$ be the set of biased features, disjoint from $\mathcal{F}_r$. If $\Phi_{\mathcal{F}_b} = 1$ and*

$$Acc_i = Acc_i^{max} - \eta \left(\sum_{f \in \mathcal{F}_b} z_{i,f} \right)$$

with η such that $\eta(i+1) - \eta(i) \geq \eta(i) - \eta(i-1)$, $\eta(0) = 0$, then any trade-off where $\exists f_1, f_2 \in \mathcal{F}_b : p_{f_1} \neq 1, p_{f_2} \neq 1$ cannot be Pareto-optimal.

Proof. Let us denote the bias intensity by $B_i = \frac{1}{k} \sum_{f \in \mathcal{F}_b} p_f$. A bias intensity of 0 indicates that no feature from $\mathcal{F}_b$ is ever selected. A bias intensity of 1 occurs when every selected feature is from $\mathcal{F}_b$. We show that, for a given bias intensity B_i , any trade-off where $\exists f_1, f_2 \in \mathcal{F}_b : p_{f_1} \neq 1, p_{f_2} \neq 1$ cannot be Pareto-optimal. We show that 1) strictly better stability and 2) better or equal average accuracy can be reached.

(1): It can be shown that $\Phi \propto \sum_{f=1}^d p_f^2$. At constant B_i , an assignment maximizes $\sum_{f \in \mathcal{F}_b} p_f^2$ if and only if it has at most one feature from $\mathcal{F}_b$ with $p_f \neq 1$. All other things being equal, *i.e.*, the selection frequencies p_f of features from $\mathcal{F}_r$, this is also true with respect to Φ .

(2): To maximize the average accuracy at constant bias intensity, the number of co-occurrences of size k has first to be minimized, then $k-1, \dots$. This is a direct consequence of the assumption that the marginal accuracy drops increase with the number of selected features from $\mathcal{F}_b$, *i.e.*, that $\eta(i+1) - \eta(i) \geq \eta(i) - \eta(i-1)$. Any assignment with at most one feature from $\mathcal{F}_b$ such that $p_f \neq 1$ does exactly that and hence maximizes accuracy for a given bias intensity B_i . $\square$

The intuition behind Theorem 1 is that forcing the selection maximizes the stability increase obtained when selecting features from the biased set a fixed amount of times while minimizing their number of co-occurrences. If the latter is representative of the accuracy, forcing the selection is the optimal strategy. Nonetheless, the accuracy assumption supposes that every feature from the biased set is equally (ir)relevant, which need not be the case, especially when $|\mathcal{F}_b|$ is large.

4.4 Conclusion

This chapter presents the hybrid-RFE algorithm. This algorithm targets stability improvements via a dedicated meta-parameter. Pareto-optimal curves

are drawn by pre-selecting some features based on a stable univariate criterion before running the multivariate recursive feature elimination (RFE) algorithm, which selects the most appropriate additional features. Increasing the number of pre-selected features improves stability at the possible cost of some prediction performance. Results on multiple microarray and mass-spectrometric data sets show that large stability increases are obtained at a small cost of classification accuracy. Using the evaluation framework from Chapter 3, one can conclude that the hybrid-RFE algorithm increases the average satisfaction from domain experts. Those increases are at least moderately statistically significant, depending on which approach the hybrid-RFE is compared to. Strong statistical significance is naturally hard to reach because of the small sizes of the data sets under study. This chapter also provides some theoretical and empirical evidence that *forcing* the selection of some stable features is an appropriate strategy.

An Importance Weighted Stability Measure | 5

This chapter defines a novel selection stability measure, which incorporates the relative importance of the selected features into the stability value. It is articulated as follows. Section 5.1 introduces the feature stability maps, a handy visualization tool for stability assessment and domain analysis. With the help of such stability maps, Section 5.2 illustrates situations where a learning algorithm is deemed stable by current measures but would still be seen as untrustworthy by domain experts. This indicates that current state-of-the-art measures are inadequate to quantify the trustworthiness of feature selection. A novel importance-weighted stability measure, Φ_{iw} , is then proposed in Section 5.3. This section also proves several good properties of Φ_{iw} and derives confidence intervals and hypothesis testing on its asymptotic value. The adequacy of the measure is first justified theoretically in Section 5.4 under the form of a property that only the measure Φ_{iw} satisfies. As Φ_{iw} depends on the importance of the selected features in predictive models, the hybrid-RFE algorithm is extended in Section 5.5 to allow the control of such importances. Using this algorithm, Section 5.6 shows empirically on real microarray and mass-spectrometric data that Φ_{iw} behaves more adequately in the context of stability optimization. Next, Section 5.7 studies the stability of standard feature selection methods according to this new index. The content of this chapter has been published in [HD21a].

5.1 Feature Stability Maps

So far in this document and in line with the literature, the Nogueira index Φ has been used for stability estimation. This index is a subset stability measure, *i.e.*, it quantifies the stability of the identity of the selected features. A predictive model is built on these selected features, and its accuracy is evaluated (see Figure 3.3). However, the contributions of each selected feature to this predictive model is not looked at. This notion of feature contribution to a predictive model is more often referred to as *feature importance*. This chapter illustrates that ignoring feature importance can have detrimental consequences when stability is optimized jointly with predictive accuracy. How to assess feature importance is studied in Section 5.3.2. In the simplest case, the importance of

a specific feature in a (generalized) linear model is directly proportional to its absolute weight value in such a model.

Before that, this section proposes a visualization tool that allows the intuitive estimation of the stability of all the learning process outcomes, which are a set of selected features *and* a predictive model built on them. An instance of these feature stability maps is depicted in Figure 5.1. Each row represents a given decision model, the whole map representing the M models learned over the M resamplings ($M = 20$ here). Each feature is assigned a color, and the horizontal extensions of the rectangles measure the importance of the features in the corresponding decision models, here linear ones estimated on each of such resamplings. For clarity, the features are sorted from left to right in decreasing order of selection frequency p_f across runs.

As a domain analysis tool, these so-called feature stability maps allow the identification of the most reliably identified features (the most frequently selected features which are put at the left of such maps) and the most important features in the predictive models (the largest rectangles) that are subsequently built on the selected features. Features combining both properties are likely to be particularly appealing for domain experts. In Figure 5.1, the green and mauve features seem particularly interesting, as they are selected in every run and also matter a lot in the predictive models. In contrast, the orange and yellow features are selected in most selection runs but are much less important in the predictive models. They look less appealing for subsequent analysis as they seem to matter less in the involved processes.

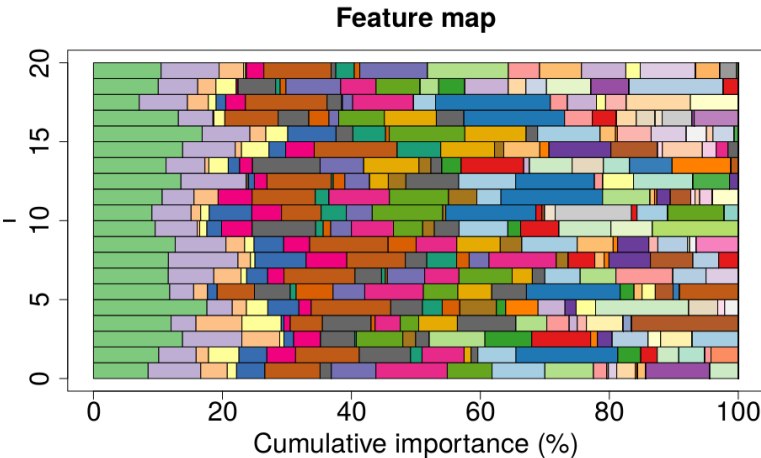


Figure 5.1: Example of feature stability map. Each row represents a given predictive model (e.g., a classifier or a regression model). Features are assigned unique colors and the horizontal extensions of the rectangles measure their importance in the models. The features are sorted from left to right in decreasing order of selection frequency p_f .

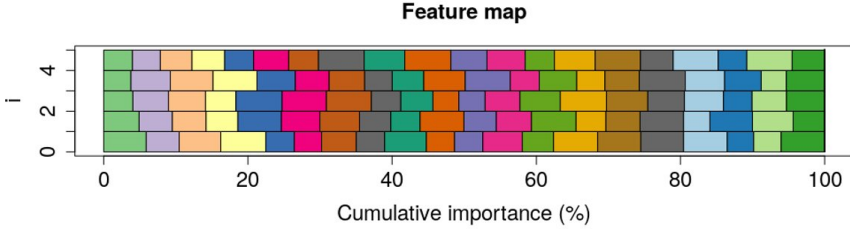


Figure 5.2: Feature stability map indicating strong interpretability. The measures Φ , Φ_{pears} and Φ_{iw} are high but the ranking measure Φ_{can} is not ($\Phi = 1$, $\Phi_{\text{pears}} = 0.97$, $\Phi_{\text{iw}} = 0.9$ and $\Phi_{\text{can}} = 0.47$).

5.2 Motivation for a new measure

The primary goal of increasing stability is the improvement of the domain experts' confidence in the learning algorithms and their outcomes. In this chapter, we propose a measure (Φ_{iw} , formally defined in Section 5.3) that weights the contributions of the selected features in the stability by their relative importance in the associated predictive models. We motivate here on several examples that such a weighted stability is beneficial for the primary goal stated above. In this section, we study the Nogueira index Φ , our proposed measure Φ_{iw} , the weight-based Pearson's correlation measure, and the ranking-based Canberra measure; both of the latter are introduced in Section 2.3.4.

A first example of a feature stability map is displayed in Figure 5.2. The learning algorithm selects the same 20 variables in each run. They are combined in a multivariate predictive model where each feature plays an approximately equal role. This map would be perfectly interpretable by domain experts. In this case, both subset-based and weight-based stability measures are very high, even equal to 1 for subset-based measures such as Φ (assuming $d > 20$ here). This is not the case for ranking stability measures, as the ranking between the 20 selected features is (by design in this toy example) random. To compute stability values, we assume that the number of input features d tends to ∞ .

Figure 5.3 illustrates a particularly interesting scenario that has been drawn from our experiments on microarray data with the hybrid-RFE. As features are sorted from left to right in decreasing selection frequency, left features are selected the most often and consequently are the most responsible for the apparent (subset-)selection stability, which is good in this example ($\Phi = 0.72$). However, it appears that the 15 features selected the most (among the 20 selected features here) have a cumulative importance of only around 25%. A domain expert would expect the most frequently selected features to be particularly useful to the task of interest (= high importance). In Figure 5.3, it appears to be the opposite, as some of them only play a marginal role in the

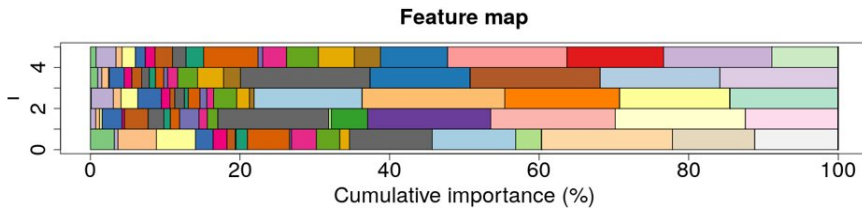


Figure 5.3: Example of a feature stability map for which Φ would be drastically overestimated. The measures Φ_{pears} and Φ_{iw} (and Φ_{can} to a lesser extent) correct this phenomenon ($\Phi = 0.72$, $\Phi_{\text{pears}} = 0.13$, $\Phi_{\text{can}} = 0.39$ and $\Phi_{\text{iw}} = 0.21$).

predictive models.

A second kind of undesirable instability is depicted in Figure 5.4. In this example, although the same subset is selected in each run, feature importance strongly varies across selection runs. Like the classical feature selection instability, a strong instability of the importance of the selected features is likely to deteriorate the interpretability of the selected features and the trust of domain experts towards their actual relevance. Subset-based stability measures are unable to grasp these nuances. Situations similar to the ones depicted in Figures 5.3 and 5.4 can naturally occur when selection stability is optimized jointly with predictive accuracy, as is studied further in Section 5.6. One can also see that Φ_{can} is higher in Figure 5.4 than in Figure 5.2, which is orthogonal to our purpose of measuring the stability of feature importance. We focus our study on subset and weight-based measures for this reason and because dedicated ranking stability measures are not designed to compare rankings of different sizes (which occurs when the number of selected features varies from run to run). For situations where domain experts would rather be interested in a feature ranking, we refer the reader to [UPC20; Jur+08; NSB17b; KV10].

As far as Figures 5.2, 5.3 and 5.4 are concerned, Φ_{pears} is able to correctly identify instability. However, we note later that Φ_{pears} lacks some important properties (see Table 6.4). Furthermore, it behaves inadequately in the setting depicted in Figure 5.5, which can naturally arise when the feature space is composed of highly correlated feature groups, as detailed in Chapter 6. In Figure 5.5, perfectly stable features have a cumulative importance of 50% while the other half of importance belongs to 5 features different in each run. Arguably, stability should be close to 0.5 in this setting which is the precise value of our proposed measure Φ_{iw} . Subset-based measures such as Φ naturally overestimate stability while the weight-based Φ_{pears} underestimates it. This phenomenon and the problems that arise from it are studied further in Chapter 6.

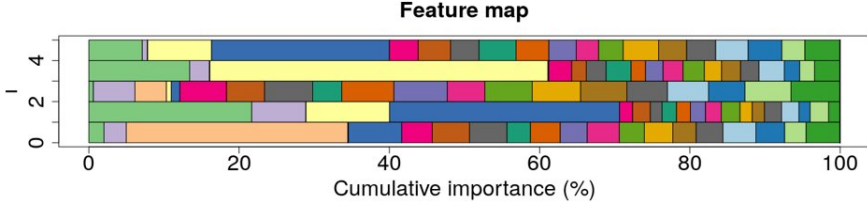


Figure 5.4: Feature stability map where feature importance is highly unstable. The subset stability Φ does not account for this instability and is maximal ($\Phi = 1$), while Φ_{can} is the highest among the maps presented here ($\Phi_{\text{can}} = 0.55$). Once again, Φ_{pears} and Φ_{iw} are able to assess this instability ($\Phi_{\text{pears}} = 0.42$, $\Phi_{\text{iw}} = 0.53$).

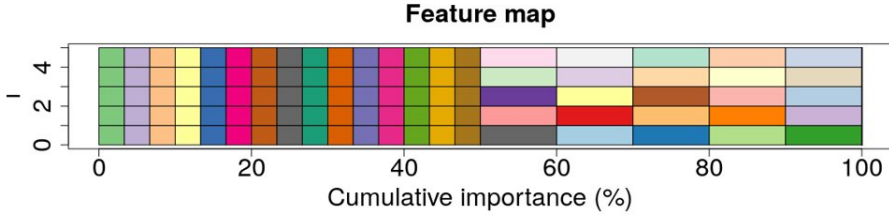


Figure 5.5: Feature stability map with half the importance space being perfectly stable while the other half is maximally unstable. The measure Φ overestimates stability as the number of stable features is high, while Φ_{pears} underestimates it ($\Phi = 0.75$, $\Phi_{\text{pears}} = 0.25$, $\Phi_{\text{can}} = 0.51$ and $\Phi_{\text{iw}} = 0.5$).

5.3 An Importance Weighted Stability Measure

This section presents a novel importance weighted stability measure Φ_{iw} . The measure is formally defined in Section 5.3.1. Section 5.3.2 proposes a new and interpretable way of computing feature importance. The properties and reductions of Φ_{iw} are studied in Section 5.3.3 and its time complexity in Section 5.3.4.

5.3.1 Definition

In this section, we extend the Kuncheva index to handle a varying number of selected features and to incorporate feature importances. We pose

$$\Phi_{\text{iw}} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} - |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}}{\bar{k} - C} \quad (5.1)$$

with

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = \sum_{f \in \mathcal{F}_i \cap \mathcal{F}_j} \min(I_{i,f}, I_{j,f}), \quad (5.2)$$

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = \frac{1}{d} \sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}), \quad (5.3)$$

and

$$C = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}, \quad (5.4)$$

$I_{i,f}$ being the importance of the selected feature f in predictive model number i . In essence, the similarity between two selection runs is defined as the sum of the common importance that selected features have between both decision models (Equation 5.2). This can be visually estimated with the help of the feature stability maps, as it corresponds to the overlap of the same colors between two corresponding rows. The overall stability value is then the average of the pairwise similarities (normalized and corrected for chance). This new stability Φ_{iw} corrects for the undesirable instability of Figures 5.3 and 5.4 as these overlaps are low in both cases. As previously stated, it is exactly equal to 0.5 in Figure 5.5 because the overlaps extend to exactly half of the feature stability map.¹

Such a stability requires an importance evaluation function: $\mathcal{I} : \{\mathcal{F}, \mathcal{P}\} \rightarrow \mathbb{R}$. This function is formally defined in Section 5.3.2. We normalize the feature importances in each selection run such that $\sum_f I_{i,f} = \bar{k}, \forall 1 \leq i \leq M$. As such a normalization would be undefined for a selection run i with $k_i = 0$ selected features, we pose $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = 0$ if $k_i = 0 \oplus k_j = 0$ and $\bar{k}$ (the average number of selected features over the M runs) if $k_i = k_j = 0$, with $\oplus$ the XOR operator.

5.3.2 Evaluating Feature Importance

The evaluation of the importance of features in a predictive model is the root of embedded feature selection algorithms. For linear models, one can use the simple function

$$\mathcal{I}_{\text{lin}}(f, \mathbf{w}) = \|\mathbf{w}\|_0 \times \frac{|\mathbf{w}_f|}{\|\mathbf{w}\|_1} \quad (5.5)$$

where $\mathbf{w}$ represents the weight vector of the model. A linear RFE builds linear models and iteratively drops the features whose importance are the lowest according to Equation (5.5). For non-linear SVM models, one can still attribute an

¹We suppose that d tends to ∞ in Figure 5.5 which here implies $C \rightarrow 0$.

importance to each feature by computing how much this feature contributes to the margin of the svm [Guy+02]. This can be done using Equation (5.6)

$$\mathcal{I}_{\text{svm}}(f, \boldsymbol{\alpha}) \propto |W^2(\boldsymbol{\alpha}) - W_{(-f)}^2(\boldsymbol{\alpha})|, \quad W_{(-f)}^2(\boldsymbol{\alpha}) = \sum_{k,l} \alpha_k \alpha_l y_k y_l (\mathbf{x}_k^{-f} \cdot \mathbf{x}_l^{-f}) \quad (5.6)$$

where $\mathbf{x}_k^{-f}$ denotes the training point k without the feature f , y_k is the label of training point k (± 1) and the α_k 's are the solutions to the svm dual problem. Similarly to the linear RFE algorithm, the non-linear svm-RFE iteratively drops features with the lowest importance according to Equation (5.6). This process produces the same ranking as Equation (5.5) for linear svms [Guy+02]. For random forest classifiers, feature importance can be measured by randomly permuting the features in the out-of-bag samples and computing the drop in predictive accuracy caused by these permutations [Bre01b; PVD+13]. For black-box classifiers (e.g., DNNs) and to provide a unified framework, we define the importance of a feature f in a predictive model p , or the sensitivity of model p to the feature f , as the inverse of the smallest noise applied to f necessary to flip the decision of model p , averaged over the n learning examples. Formally,

$$\mathcal{I}_n(f, p) \triangleq \frac{1}{n} \sum_{i=1}^n \frac{k_p \times \mathcal{I}(f, p, \mathbf{x}_i)}{\sum_{f' \in \mathcal{F}_p} \mathcal{I}(f', p, \mathbf{x}_i)}, \quad \mathcal{I}(f, p, \mathbf{x}_i) = \frac{\sigma_f}{\delta_{\mathbf{x}_i, p, f}} \quad (5.7)$$

where k_p is the number of features used by model p , $\delta_{\mathbf{x}_i, p, f}$ is the smallest additive change (in absolute value) required to feature f such that the decision of the predictive model p on example $\mathbf{x}_i$ changes, and σ_f the standard deviation of feature f .

Intuitively, if one can change feature f by large amounts without perturbing the decisions of the model (here thought as a classifier), then f is not important in the decisions. On the contrary, if a small change to f causes many decision switches, then the model is highly sensitive to it. Theorem 2 states that computing feature importance using Equation (5.5) or (5.7) for linear models is equivalent when the selected features are normalized to unit variance.

Theorem 2. *For a linear decision model p with weights $\mathbf{w}$, evaluated from n learning examples with k_p features normalized to unit variance,*

$$\mathcal{I}_{\text{lin}}(f, \mathbf{w}) = \|\mathbf{w}\|_0 \times \frac{|\mathbf{w}_f|}{\|\mathbf{w}\|_1} = \mathcal{I}_n(f, p).$$

Proof. As features are normalized to unit variance, $\sigma_f = 1$, $\forall f$. The decision function of a linear model p with weights $\mathbf{w}$ can be written $D(\mathbf{w}, \mathbf{x}) =$

$\text{sign}(\sum_{f=1}^{k_p} w_f \times x_f + w_0)$. The smallest change $\delta_{x_i,p,f}$ to a feature f required to flip the decision of data point $\mathbf{x}_i$ is the change required such as to make $D(\mathbf{w}, \mathbf{x}_i) = 0$. Then,

$$\begin{aligned} \delta_{x_i,p,f} &= \frac{D(\mathbf{w}, \mathbf{x}_i)}{|w_f|} \Rightarrow \mathcal{I}(f, p, \mathbf{x}_i) = \frac{|w_f|}{D(\mathbf{w}, \mathbf{x}_i)} \\ &\Rightarrow \mathcal{I}_n(f, p) = \frac{1}{n} \sum_{i=1}^n \frac{k_p \times \frac{|w_f|}{D(\mathbf{w}, \mathbf{x}_i)}}{\sum_{f' \in \mathcal{F}_p}^d \frac{|w_{f'}|}{D(\mathbf{w}, \mathbf{x}_i)}} \\ &\Rightarrow \mathcal{I}_n(f, p) = \frac{1}{n} \sum_{i=1}^n k_p \times \frac{|w_f|}{\|\mathbf{w}\|_1} = k_p \times \frac{|w_f|}{\|\mathbf{w}\|_1} \triangleq \|w_0\| \times \frac{|w_f|}{\|\mathbf{w}\|_1} \triangleq \mathcal{I}_{lin}(f, \mathbf{w}). \end{aligned}$$

□

With feature importance defined by Equation (5.7), computing it for all M selection runs can be done in $O(M\bar{k})$ for linear models (according to Theorem 2) and in $O(M\bar{k}n)$ in the non-linear case. Note that for data sets with a high number of samples, feature importance can be evaluated on a fraction of the data if the $O(M\bar{k}n)$ complexity becomes prohibitive.

5.3.3 Properties and Reductions

In this section, we show that our proposed stability measure Φ_{iw} satisfies the following properties, adapted from [NSB17a]. A new desirable property concerning stability optimization is then defined in Section 5.4. Chapter 6 also introduces a new property in the context of feature correlations. Both properties are proved for Φ_{iw} in appendix.

Property 1. *Fully defined: the measure is defined for every possible importance combinations.*

Property 2. *Maximum stability $\Leftrightarrow$ deterministic importance: the measure reaches its maximum if and only if the importance of every feature is constant across all runs.*

Property 3. *Bounds: the measure is bounded by constants not dependent on the overall number of features d or on the average number of features selected $\bar{k}$.*

Property 4. *Correction for chance: the measure is constant in expectation (here set to 0) when features are selected randomly.*

The importance weighted stability measure Φ_{iw} is fully defined: it is defined everywhere except when no feature is ever selected, or every feature is always selected with equal importance. In both cases, one can hardly speak of feature selection. The measure is maximal whenever the same feature subset is always selected and the importances of the selected features are constant across runs. It is lower bounded by $\frac{-1}{M-1}$ and upper bounded by 1. As the number of runs M is greater than or equal to 2, Φ_{iw} is always bounded by -1 and 1 , which is necessary for relevant comparisons, and is asymptotically lower bounded by 0 as M tends to ∞ . The measure is also corrected for chance as its expected value is constant (here set to 0) whenever features are selected randomly. These properties are proved in Appendix A.1.

Theorem 3, proved in Appendix A.2, shows more clearly the similitude between Φ_{iw} and the Kuncheva index whenever the importance of all selected features is evenly distributed between them in any given run.

Theorem 3. *Whenever the importance of all selected features is evenly distributed between them in any given run,*

$$\Phi_{iw} = \frac{\mu_M \left[\frac{\bar{k} |\mathcal{F}_i \cap \mathcal{F}_j|}{\max(k_i, k_j)} \right] - \frac{\bar{k}}{d} \mu_M \left[\frac{k_i k_j}{\max(k_i, k_j)} \right]}{\bar{k} - \frac{\bar{k}}{d} \mu_M \left[\frac{k_i k_j}{\max(k_i, k_j)} \right]}$$

with $\mu_M(g(i, j)) = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M g^*(i, j)$, $g^*(i, j) = 1$ if $k_i = k_j = 0$, $g(i, j)$ otherwise.

The correction for chance term of the Kuncheva index, $\frac{\bar{k}^2}{d}$, is extended here to $\frac{\bar{k}}{d} \mu_M \left[\frac{k_i k_j}{\max(k_i, k_j)} \right]$ to handle a varying number of selected features. Also, the subset intersection between selection runs i and j , $|\mathcal{F}_i \cap \mathcal{F}_j|$, is weighted by the term $\frac{\bar{k}}{\max(k_i, k_j)}$, such that selection runs with a high or low number of selected features influence the overall stability value by the same amount. Chapter 6 shows that this property is convenient when dealing with groups of correlated variables.

Theorem 4, proved in Appendix A.2 as well, further shows that whenever the number of selected features is constant across runs, our proposed measure degenerates into the Kuncheva index and, thus, into the Nogueira index Φ .

Theorem 4. *Whenever the importance of all selected features is evenly distributed between them in any given run and the number of selected features is constant across the M runs,*

$$\Phi_{iw} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j| - \frac{\bar{k}^2}{d}}{\bar{k} - \frac{\bar{k}^2}{d}},$$

which is the usual expression of the Kuncheva index.

Theorem 3 and Theorem 4 indicate that, when the predictive models approximately use their features equally, our measure Φ_{iw} behaves in a similar manner to existing measures (as is validated in Section 5.7). However, we show in Sections 5.4 and 6.3.2 that current measures are vulnerable to undesirable behaviors in certain situations, while our measure is more robust in this regard.

We further show in Appendix A.2 that the measure Φ_{iw} can be re-stated in a frequency-based form, as

$$\Phi_{\text{iw}} = 1 - \frac{\frac{M}{M-1}(\bar{k} - \sum_{f=1}^d I_f^* p_f^2)}{\bar{k} - C} = 1 - \frac{\sum_{f=1}^d I_f^* s_f^2 + \frac{M}{M-1}(\bar{k} - \sum_f I_f^* p_f)}{\bar{k} - C} \quad (5.8)$$

with a properly normalized global feature importance $I_f^* = \sum_{i,j=1}^M \frac{\min(I_{i,f}, I_{j,f})}{|z_{f,\cdot} \neq 0|^2}$, where $|z_{f,\cdot} \neq 0|$ is the number of runs where feature f is selected. This new formulation makes even more explicit the corrections for the instabilities of Figure 5.3 and 5.4: the selection variance s_f^2 of features with a high importance I_f^* accounts more in the overall stability such that the overestimation of stability, when frequently selected features are not used for prediction, is corrected (Figure 5.3). Furthermore, having features with highly varying importances in different selection runs is penalized by the term $(\bar{k} - \sum_f I_f^* p_f)$ which is equal to zero only when $I_{i,f} = I_f^*, \forall i$.

5.3.4 Time Complexity

This section studies the time complexity needed to compute our novel index Φ_{iw} .

In particular, the corrective term C in the definition of Φ_{iw} (Equation 5.1) can be computed in $O(M^2 \bar{k} + M \bar{k} \log(k))$ time by Algorithm 4, with $\bar{k} \log(k) = \frac{1}{M} \sum_{i=1}^M k_i \log(k_i)$, assuming that the feature selection algorithm produces unsorted feature importances.² The overall time complexity to compute Φ_{iw} from feature importance is also $O(M^2 \bar{k} + M \bar{k} \log(k))$, as computing the pairwise intersections $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}$ requires only $O(M^2 \bar{k})$.

According to the definition of feature importance proposed in Section 5.3.2, such feature importance can be computed in $O(M \bar{k} n)$. This quantity reduces to $O(M \bar{k})$ for linear models according to Theorem 2. Consequently, the time complexity of the joint process of evaluating feature importance and computing stability is $O(M^2 \bar{k} + M \bar{k} \log(k) + M \bar{k} n)$ in general and $O(M^2 \bar{k} + M \bar{k} \log(k) + M \bar{k}) = O(M^2 \bar{k} + M \bar{k} \log(k))$ when dealing with linear predictive models.

²Otherwise, the time complexity becomes $O(M^2 \bar{k})$.

Algorithm 4 Computing C .

```

1: procedure  $C(\mathcal{F}, I, M)$   $\triangleright O(M^2\bar{k} + Mk \log(k))$ 
2:   for  $i \in 1:M$  do  $\triangleright O(Mk \log(k))$ 
3:      $I_i \leftarrow \text{sort}(I_i)$   $\triangleright O(k_i \log(k_i))$ 
4:      $Isum_i[l] \leftarrow \sum_{p \leq l} I_i[p] \quad \forall 1 \leq l \leq k_i$   $\triangleright O(k_i)$ 
5:    $C \leftarrow 0$ 
6:   for  $i, j \in 1:M$  do  $\triangleright O(M^2\bar{k})$ 
7:      $C \leftarrow 0, |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} \leftarrow 0, c_i = 0$ 
8:     for  $c_j \in 1:k_j$  do  $\triangleright O(k_i + k_j)$ 
9:       while  $c_i < k_i \ \& \ I_i[c_i + 1] < I_j[c_j]$  do
10:         $c_i \leftarrow c_i + 1$ 
11:       if  $c_i == 0$  then
12:         $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} += I_j[1] \times k_i$ 
13:       else
14:        if  $c_i == k_i$  then
15:          $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} += Isum_i[k_i]$ 
16:        else
17:          $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} += Isum_i[c_i] + (k_i - c_i) \times I_j[c_j]$ 
18:        $C \leftarrow C + |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} / M^2$ 
19:   return  $\frac{C}{d}$ 

```

The computational complexities are reported in the comments of the main steps of this algorithm. It first sorts the selected features by increasing importance for each selection run and stores the cumulative importance of the first l features in $Isum_i[l]$. Then, for each pair of runs, the quantity $\sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'})$ can be computed by traversing each sorted array of features once. The invariant of the `for` loop on line 8 is the following:

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = \sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j[1:(c_j-1)]} \min(I_{i,f}, I_{j,f'}).$$

After the `while` loop on line 9 is executed, c_i is equal to the biggest importance in run i lower than $I_j[c_j]$, the importance of the feature of run j considered at this iteration of the `for` loop. The sum of minima

$$\sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j[c_j]} \min(I_{i,f}, I_{j,f'})$$

can then be decomposed into two terms (the cases where $c_i = 0$ or $c_i = k_i$ are handled explicitly in the algorithm):

$$\begin{aligned} & \sum_{f \in \mathcal{F}_i[1:c_i], f' \in \mathcal{F}_j[c_j]} \min(I_{i,f}, I_{j,f'}) + \\ & \sum_{f \in \mathcal{F}_i[(c_i+1):k_i], f' \in \mathcal{F}_j[c_j]} \min(I_{i,f}, I_{j,f'}) = Isum_i[c_i] + (k_i - c_i)I_j[c_j]. \end{aligned}$$

5.3.5 Sampling Distribution And Statistical Tools

The Nogueira index Φ weakly tends to a normal distribution. This allows the derivation of approximate confidence intervals, as introduced in Section 3.4.3. From their previous work [NSB17a], we derive the sampling distribution of the Kuncheva Index and extend it to our measure Φ_{iw} . Statistical tools, such as confidence intervals and hypothesis testing frameworks, are constructed from this sampling distribution. These tools are then validated experimentally on real data.

By particularizing Equation 3.19 and 3.20 to the case where $k_i = k \forall i$ (when Φ reduces to the Kuncheva index), we get³, as $M \rightarrow \infty$,

$$T_{\text{ku}} = \frac{\hat{\Phi} - \Phi}{\sqrt{v(\hat{\Phi})}} \rightarrow \mathcal{N}(0, 1), \text{ where } v(\hat{\Phi}) = \frac{4}{M^2} (\hat{\Phi}_{(i)} - \hat{\Phi}_{(\cdot)})^2,$$

³A lot of terms can be discarded in Equation 3.20 when $k_i = k \forall i$ as they appear in both $\hat{\Phi}_{(i)}$ and $\hat{\Phi}_{(\cdot)}$. We chose to keep the $\frac{k^2}{d}$ term in the numerator for interpretability. Also note that $\sum_{j=1}^M z_{jf} p_f = \frac{1}{M} \sum_{j=1}^M |\mathcal{F}_i \cap \mathcal{F}_j|$ when $M \rightarrow \infty$ and $k_i = k \forall i$.

$$\hat{\Phi}_{(i)} = \frac{\frac{1}{M} \sum_{j=1}^M |\mathcal{F}_i \cap \mathcal{F}_j| - \frac{k^2}{d}}{k - \frac{k^2}{d}} \text{ and } \hat{\Phi}_{(\cdot)} = \frac{1}{M} \sum_{i=1}^M \hat{\Phi}_{(i)}. \quad (5.9)$$

The value $\hat{\Phi}_{(i)}$ estimates the average similarity between selection run i and the population of M runs. We here extend Equation 5.9 to Φ_{iw} and experimentally validate it on real data for a wide range of scenarios. Equation 5.9 becomes

$$T_{\text{iw}} = \frac{\hat{\Phi}_{\text{iw}} - \Phi_{\text{iw}}}{\sqrt{v(\hat{\Phi}_{\text{iw}})}} \rightarrow \mathcal{N}(0, 1), \text{ where } v(\Phi_{\text{iw}}) = \frac{4}{M^2} (\hat{\Phi}_{(i)}^{\text{iw}} - \hat{\Phi}_{(\cdot)}^{\text{iw}})^2,$$

$$\hat{\Phi}_{(i)}^{\text{iw}} = \frac{\frac{1}{M} \sum_{j=1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} - C}{\bar{k} - C} \text{ and } \hat{\Phi}_{(\cdot)} = \frac{1}{M} \sum_{i=1}^M \hat{\Phi}_{(i)}. \quad (5.10)$$

A $(1 - \alpha)\%$ confidence interval for Φ_{iw} is then

$$[\hat{\Phi}_{\text{iw}} - z_{1-\frac{\alpha}{2}}^* \sqrt{v(\hat{\Phi}_{\text{iw}})}, \hat{\Phi}_{\text{iw}} + z_{1-\frac{\alpha}{2}}^* \sqrt{v(\hat{\Phi}_{\text{iw}})}] \quad (5.11)$$

where $z_{1-\frac{\alpha}{2}}^*$ is the inverse cumulative of a standard normal distribution at $1 - \frac{\alpha}{2}$.

Hypothesis testing can also be performed to assess the difference between the stability of two feature selection algorithms with estimates $\hat{\Phi}_1$ and $\hat{\Phi}_2$ using the statistic

$$T_{\text{comp}} = \frac{\hat{\Phi}_1 - \hat{\Phi}_2}{\sqrt{v(\hat{\Phi}_1) + v(\hat{\Phi}_2)}}. \quad (5.12)$$

Whenever $T_{\text{comp}} > z_{1-\frac{\alpha}{2}}^*$, the null hypothesis that $\Phi_1 = \Phi_2$ can be rejected.

Experimental validation is performed with the logistic-RFE and the lasso feature selection algorithms, applied to three data sets with various $\frac{n}{d}$ ratios and stability values. These scenarios are reported in Table 5.1. The number of selected feature is set to $\min(20, \sqrt{d})$. The true stability values are computed with $M = 50000$.

Algo.	Φ	Φ_{iw}	Algo.	Φ	Φ_{iw}	Algo.	Φ	Φ_{iw}
log-RFE	0.42	0.28	log-RFE	0.73	0.51	log-RFE	0.73	0.72
lasso	0.22	0.17	lasso	0.43	0.40	lasso	0.73	0.75

(a) **alon** - low stability:
n=62, d=2000, $\bar{k}=20$

(b) **lsvt** - intermediate stability:
n=126, d=310, $\bar{k}=18$

(c) **iono** - high stability:
n=350, d=34, $\bar{k}=6$

Table 5.1: Scenarios used for the validation of the sampling distribution and statistical tools.

The meta-parameters of the logistic-RFE are set such as to promote variety among the importances of the selected features. This allows us to validate the confidence intervals in situations where Φ_{iw} departs from the Kuncheva index. A typical feature stability map of the logistic-RFE obtained on **alon** with the chosen meta-parameters is displayed in Figure 5.6a. The lasso is used to extend our analysis to situations where the number of selected features varies from run to run. Figure 5.6b depicts a typical feature stability map of the lasso obtained on **lsvt**.

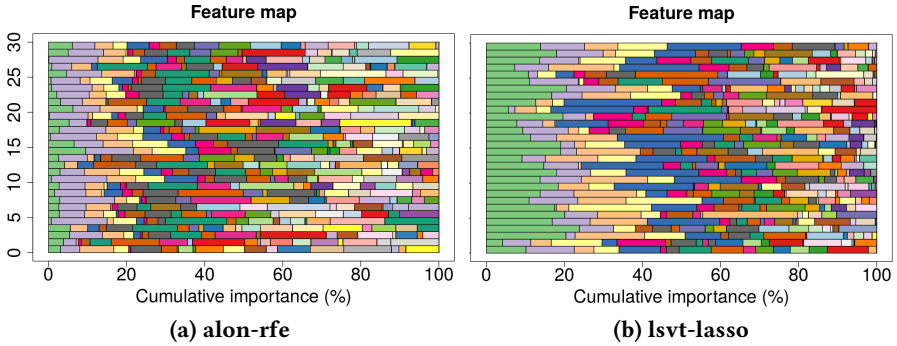


Figure 5.6: Feature stability maps obtained with the logistic-RFE on **alon (a) and with the lasso on **lsvt** (b).**

Figure 5.7 shows the QQ-plots of T_{iw} and the standard normal distribution $\mathcal{N}(0, 1)$ when $M = 4$ and $M = 100$. The selection algorithm is the logistic-RFE, applied on the **alon** data set. Clearly, the T-statistic is not well normally distributed on the edges of the distribution when $M = 4$. Values much lower and higher than expected are sometimes taken by T_{iw} . When $M = 100$, the quantiles of T_{iw} and those of a standard normal distribution match nicely, providing empirical evidence for Equation 5.10.

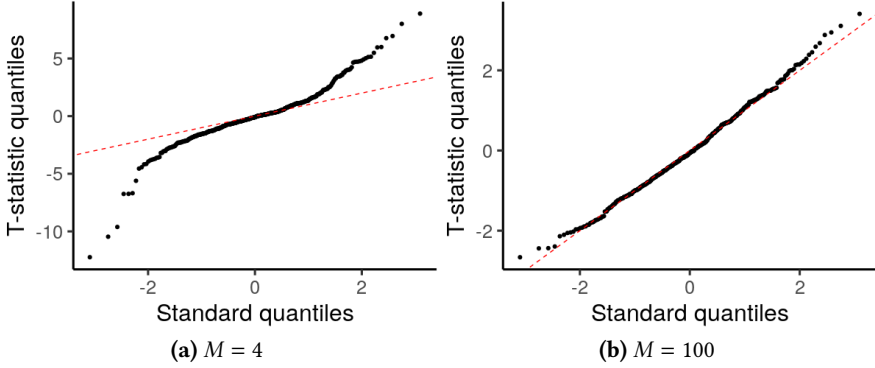


Figure 5.7: QQ-plot of the T-statistic (T_{iw} , Equation 5.10) quantiles versus the standard normal quantiles for $M = 4$ (a) and $M = 100$ (b). The selection method is the logistic-RFE, applied on the `alon` data set. The red dashed line is the diagonal line, i.e., what is expected in case of perfect normality (and infinite sample size, which is here set to 500).

Figure 5.8 depicts the coverage probabilities of the confidence intervals on Φ_{iw} as a function of the number of selection runs M . These coverage probabilities quickly converge towards $1 - \alpha = 0.9$ when M increases in all cases. Figure 5.9 studies the type I error of the hypothesis test based on the statistic T_{comp} (Equation 5.12). The type I error is the fraction of time the null hypothesis $\Phi_1 = \Phi_2$ is rejected even though it is true. This error converges towards $\alpha = 0.1$, with the exception of the lasso on `1svt`, which nonetheless remains within the statistically acceptable range.

5.4 Importance Weighted Decomposition

The hybrid-RFE, introduced in Section 4.1, combines the selection of N features, which are first chosen based on a univariate criterion, and the selection of $k - N$ features, which are then found, in a multivariate fashion, by Algorithm 3. In this section, we generalize this idea and provide a theoretical analysis of the evolution of the stability when Q selection methods are combined ($Q = 2$ for the hybrid-RFE). From this analysis, it follows that, unlike the other measures, the stability Φ_{iw} is particularly suitable to be optimized along with predictive accuracy.

Consider the following scenario. For each selection run m , we learn Q independent linear models on non-overlapping selected feature sets $\mathcal{F}_q(m)$: $\sum_{f \in \mathcal{F}_q(m)} w_f(m) \times x_f \geq t_q(m)$, with the normalization $\sum_{f \in \mathcal{F}_q(m)} w_f(m) = \bar{k}_q$. Each of the Q models is found by a given selection method. The overall prediction model is defined as a fixed linear combination of the Q linear models: $\sum_q \delta I_q \sum_{f \in \mathcal{F}_q(m)} w_f(m) \times x_f \geq t(m)$. The parameter δI_q modulates the im-

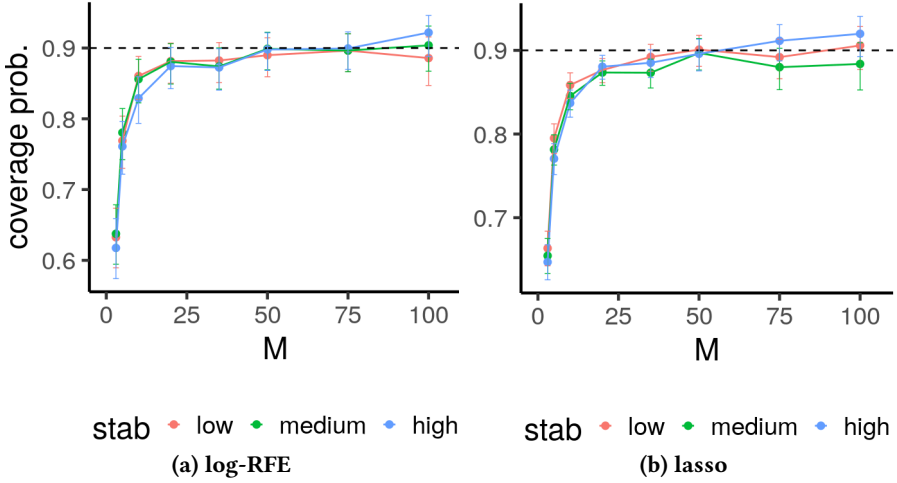


Figure 5.8: Coverage probabilities of the confidence intervals on Φ_{iw} , as a function of the number of selection runs M for three data sets with various $\frac{n}{d}$ ratios and stability values (see Table 5.1).

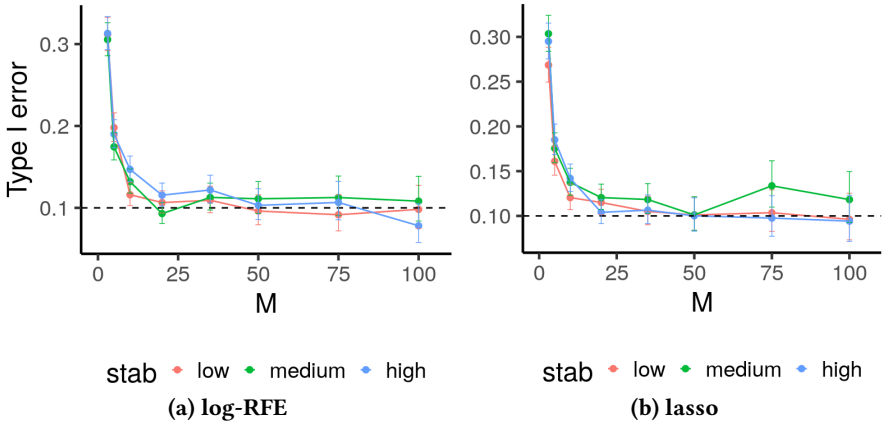


Figure 5.9: Type I error of the hypothesis test based on the statistic T_{comp} (Equation 5.12) in the three scenarios presented in Table 5.1.

portance of the selected features from particular selection methods. In this scenario, Property 5 states that it should be possible to decompose stability in multiple terms, respectively capturing the stability of each selection method, noted here Φ_{iw}^q . These terms are weighted by $\frac{\bar{k}_q}{\bar{k}} \delta I_q$, such that a selection method q has more weight in the overall stability when it is responsible for a large fraction of the selected features and when these features are important in the combined predictive model. This property is proved for Φ_{iw} in Appendix A.1 under the assumption that d tends to ∞ .

Property 5. *Importance weighted decomposition: when combining the non overlapping selected feature sets of Q different methods, which produce Q models, in a single predictive model, if features selected by method q have their importance multiplied by δI_q in the combined predictive model of each selection run, stability can be expressed as a weighted sum of the Q prior stabilities:*

$$\Phi_{\text{iw}} = \sum_{q=1}^Q \frac{\bar{k}_q}{\bar{k}} \delta I_q \Phi_{\text{iw}}^q$$

with Φ_{iw}^q the stability of method q alone, and δI_q the factor such that $I_{i,f} = \delta I_q \times I_{i,f}^q$.

As far as the hybrid-RFE is concerned, whenever the N features of the stable set are less relevant, combining both selections reduces their importance in the predictive models and increases the importance of the $k - N$ multivariate features ($\delta I_{q=\text{univariate}} < 1$ and $\delta I_{q=\text{RFE}} > 1$). This gives more weight to $\Phi_{\text{iw}}^{q=\text{RFE}}$ in Property 5, thus limiting the stability increase provided by the forced univariate selection of the N features.

Section 4.2 optimizes jointly the measure Φ with the predictive accuracy and shows that the quality of the reachable compromises is highly dependent on the stability of the stable set rather than on its predictive performance. In general, better compromises are obtained when the N pre-selected features are stable, even if not relevant to the prediction task. This is caused by the fact that, unlike Φ_{iw} , the selection of stable yet marginally important features increases Φ . Indeed, a similar result to Property 5 can be derived for Φ when d tends to ∞ :

$$\Phi = \sum_{q=1}^Q \frac{\bar{k}_q}{\bar{k}} \Phi^q, \quad (5.13)$$

which does not depend on δI_q . Consequently, a good (accuracy, Φ) compromise is not necessarily meaningful, as Φ could have been artificially increased by the selection of stable features that marginally take part in the predictive models. Subset measures, in general, can not be optimized soundly with predictive accuracy, as is shown experimentally in Section 5.6.

The measure Φ_{pears} can not be decomposed in multiple terms under the scenario described by Property 5. Still, if we assume that the q methods distribute importance evenly among their respective selected features (*i.e.*, $I_{f,i}^q = 1, \forall f, i, q$), the following decomposition holds when d tends to ∞ :

$$\Phi_{\text{pears}} = \frac{\sum_{q=1}^Q \delta I_q^2 \bar{k}_q \Phi_{\text{pears}}^q}{\sum_{q=1}^Q \delta I_q^2 \bar{k}_q}. \quad (5.14)$$

In this specific case, the relative contributions of the prior stabilities Φ_{pears}^q to the combined stability Φ_{pears} are proportional to δI_q^2 . Due to this δI_q^2 factor, the stability Φ_{pears} , like Φ_{iw} , can not be increased by the selection of stable, yet marginally important features. However, as shown in Section 6.3.2 and on the illustrative example below, this quadratic dependency can have some negative consequences.

Consider the following example with $Q = 2$. Both selection methods distribute importance evenly among their respective selected features. The first method always selects the same 15 features and is thus perfectly stable: $\Phi^1 = \Phi_{\text{pears}}^1 = \Phi_{\text{iw}}^1 = 1$. The second method always selects five features, but these five features never overlap across the M selection runs. Method 2 is perfectly unstable: $\Phi^2 = \Phi_{\text{pears}}^2 = \Phi_{\text{iw}}^2 = 0$, when d tends to ∞ . Assume that we combine these two methods and obtain the feature stability map depicted in Figure 5.5. This map is obtained with $\delta I_1 = \frac{2}{3}$ and $\delta I_2 = 2$. In this scenario, according to Equation (5.13), $\Phi = \frac{15}{20} \times \Phi^1 + \frac{5}{20} \times \Phi^2 = 0.75$. According to Equation (5.14), $\Phi_{\text{pears}} = \frac{1}{4} \times \Phi_{\text{pears}}^1 + \frac{3}{4} \times \Phi_{\text{pears}}^2 = 0.25$. Finally, according to Property 5, $\Phi_{\text{iw}} = \frac{1}{2} \times \Phi_{\text{iw}}^1 + \frac{1}{2} \times \Phi_{\text{iw}}^2 = 0.5$, which is the preferable stability value for such a feature stability map.

5.5 The hybrid-RFE Extended

This section proposes an extension to the hybrid-RFE, which aims to increase stability by modulating the importance of the selected features. This extension estimates which selected features are likely to be frequently selected across the M resampling runs. It then applies differential shrinkage on them (in the last stage of the algorithm, *i.e.*, at line 10 of Algorithm 3) to increase their importance in the decision models. From the left-hand side of Equation (5.8), repeated here for convenience:

$$\Phi_{\text{iw}} = 1 - \frac{\frac{M}{M-1} (\bar{k} - \sum_{f=1}^d I_f^* p_f^2)}{\bar{k} - C},$$

it follows that increasing I_f^* of frequently selected features (features with a high p_f) increases stability. In order to increase $I_f^* = \frac{1}{M^2} \sum_{i,j}^M \min(I_{i,f}, I_{j,f})$,

one must increase the importance $I_{i,f}$ of feature f in multiple selection runs jointly.

As RFE drops features iteratively, it is possible to measure, for each feature, how close they are to the elimination during the selection process. We define an overall frequency score, which estimates the true selection frequency of the selected features

$$sc_f = \frac{|\mathcal{F}_0|}{|\mathcal{F}_\#|} \prod_{s=1}^{\# \text{ pruning steps}} \frac{|\mathcal{F}_s| - r_f(s) + 1}{|\mathcal{F}_{s-1}| - r_f(s) + 1}, \quad (5.15)$$

with $\mathcal{F}_s$ the set of selected features after s RFE pruning steps ($|\mathcal{F}_0| = d$ and $|\mathcal{F}_\#| = k$) and $r_f(s)$ the rank of feature f at step s (RFE ranks features at each iteration and drops a fraction of the least relevant ones). Features close to the elimination at multiple steps (*i.e.*, features for which $r_f(s)$ is often close to $|\mathcal{F}_s|$) have low scores while consistently high-ranked features have high scores. This score has the convenient properties of being bounded between 0 and 1, and to be independent of the number of pruning steps. Indeed, assuming that a feature f has a constant ranking $r_f(s) = r_f \leq k$, then

$$sc_f = \frac{|\mathcal{F}_0|}{|\mathcal{F}_\#|} \times \frac{|\mathcal{F}_\#| - r_f + 1}{|\mathcal{F}_0| - r_f + 1},$$

which does not depend on the number of steps.

Figure 5.10 depicts the correlation between the selection frequency p_f of feature f , across $M = 100$ selection runs and the average frequency score sc_f given by Equation (5.15), on two typical microarray data sets (singh (left) and chiaraetti (right) that are introduced in Section 4.2). The average frequency score sc_f is computed only over the runs for which f is selected (the score is not defined for the other runs). Features that are selected more often tend to have higher frequency scores (the (Pearson) correlations between both variables are 0.68 and 0.54 for singh and chiaraetti, respectively). We then apply the following regularization in order to reduce the importance of the selected features with a low frequency score sc_f :

$$\mathcal{R} = \lambda_f * (1 + \alpha * (1 - sc_f)) \| \mathbf{w} \|_2, \quad (5.16)$$

with α a meta-parameter determining the amplitude of the differential shrinkage. This is done in the last stage of the hybrid-RFE algorithm (line 10 of Algorithm 3). Features with a low frequency score are more regularized which is expected to decrease their weight and thus their importance in the linear predictive models.

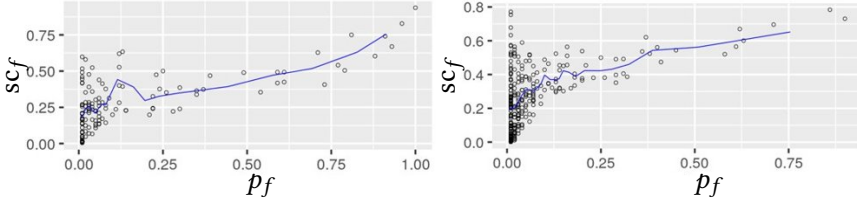


Figure 5.10: Correlation between the selection frequency p_f of feature f with its average frequency score sc_f on **singh** (left) and **chiaretti** (right).

5.6 Case Study: Decision Making for Cancer Diagnosis

This section studies the compromises between accuracy and stability that are achievable with the hybrid-RFE on five microarray and one mass spectrometric data sets. We show here that the choice of the stability measure heavily influences decision-making. In particular, we show that the Pareto-optimal front differs for each measure and that optimizing Φ or Φ_{pears} can lead to unsatisfactory feature stability maps. Furthermore, optimizing these measures usually gives a false sense of stability, further hindering appropriate decision-making. We use the same data sets as in Chapter 4 (Table 4.1) except for **chin** for which no concrete compromise can be made (as shown in Chapter 4). The feature space is again first pre-filtered by removing the features with the lowest variance. The amount of pre-filtering is found such as to maximize the predictive performance of the classical RFE ($N = 0$) and is kept constant for all experiments. To measure the accuracy and stability obtained with a given set of meta-parameters, we use the bootstrap protocol depicted in Figure 3.3. This protocol draws with replacement M samples of the same size as the original data set. Each model is evaluated on the out-of-bag examples, and the mean classification accuracy is reported. The selection stability is evaluated using Equation (2.37)(Φ), (2.39)(Φ_{pears}) and (5.1)(Φ_{iw}), over the $M = 100$ resamplings.

We perform experiments using the hybrid-RFE with the additional meta-parameter α , introduced in Section 5.5. The N pre-selected features are ranked according to the considered univariate criterion. They are put, in that order, on top of the RFE ranking at each iteration, such that their frequency score sc_f given by Equation (5.15) is the highest. Increasing α is thus expected to increase the importance of these N pre-selected features in the predictive models, as they are less regularized (Equation 5.16).

We study in Figures 5.11 and 5.12 the achievable compromises in the (accuracy, stability) objective space using the hybrid-RFE on the **singh** data set. The star in each figure represents a typical trade-off obtained with the classic RFE ($N = 0$), which is here slightly off the Pareto-curve. The plain points represent all Pareto-optimal trade-offs when Φ (red), Φ_{pears} (green) and Φ_{iw} (blue) are used to assess stability. They are obtained with the hybrid-RFE with dif-

ferent sets of meta-parameters. These plain points are the same across the three subfigures. Our measure Φ_{iw} can be increased much less than Φ and Φ_{pears} . The three measures are provably equivalent whenever the importance of the selected features are always 1. A strong difference among the values taken by Φ , Φ_{pears} and Φ_{iw} suggests that the selection moves away from this base scenario. We illustrate this phenomenon in the rest of this section. Most importantly, we show that optimizing Φ or Φ_{pears} respectively leads to situations where $\Phi \gg \Phi_{iw}$ and $\Phi_{pears} \gg \Phi_{iw}$, with poorly interpretable feature stability maps.

To this goal, consider the circled points in Figures 5.11 and 5.12. They are the compromises $(\tilde{a}_b, \Phi)$ (red), $(\tilde{a}_b, \Phi_{pears})$ (green) and $(\tilde{a}_b, \Phi_{iw})$ (blue) which are obtained with meta-parameters on the $(\tilde{a}_b, \Phi_-)$ Pareto-optimal curve ($\Phi_- = \Phi$ (Figure 5.11), $\Phi_- = \Phi_{pears}$ (Figure 5.12a) and $\Phi_- = \Phi_{iw}$ (Figure 5.12b)). For instance, the blue circled points in Figure 5.11 are the compromises in the $(\tilde{a}_b, \Phi_{iw})$ objective space that correspond to the $(\tilde{a}_b, \Phi)$ Pareto-optimal curve. These blue circled points depart from the $(\tilde{a}_b, \Phi_{iw})$ Pareto-optimal curve which consists of the plain blue points. This implies that the Pareto-optimal curves of Φ and Φ_{iw} are obtained with different meta-parameters choices, otherwise the two blue curves would coincide in Figure 5.11. Figure 5.11 also illustrates that increasing Φ is not guaranteed to increase Φ_{iw} or Φ_{pears} , as the blue and green circled curves sometimes move back towards lower stability values. The feature stability maps annotated to Figure 5.11 clearly show that optimizing Φ tends to reduce the importance of frequently selected features in the predictive models. In other words, Φ is best increased here by the selection of stable features, yet marginally used for prediction. The Pareto-optimal curve is drawn mostly by gradually increasing the number of pre-selected features N .

Figure 5.12a shows that maximizing Φ_{pears} is not guaranteed to increase Φ or Φ_{iw} . Optimizing Φ_{pears} tends to give large yet highly varying importances to frequently selected features, which also hinders sound domain analysis. The stability Φ_{pears} is largely improved by raising the value of α_f well above 0. This gives more importance to the most frequently selected features (see Section 5.5), but, as a side-effect, these importances strongly vary from run to run. This phenomenon is not penalized appropriately by the measure Φ_{pears} .

Finally, optimizing Φ_{iw} provides a much nicer feature stability map where features have reasonably constant importance across runs. Increasing Φ_{iw} is much more difficult as it does not suffer from the two previously mentioned pitfalls. Yet, it can still be slightly increased by forcing the selection of some features ($N > 0$) and increasing the final regularization parameter λ_f . This dependence on λ_f is studied in more detail in Section 5.7.3 for the classic RFE algorithm. Applying some differential shrinkage with the parameter α_f can also be slightly beneficial, but to a much lower extent than it is for Φ_{pears} .

The choice of the stability measure also influences the predictive performance of the chosen compromise. Let us assume that a domain expert has Equation 3.11 as satisfaction function, with preference parameter $\gamma \rightarrow 0$. Stability is considered extremely important by this expert. On `singh`, for $\gamma \rightarrow 0$, the chosen compromises correspond to the feature stability maps to the bottom of Figure 5.11 and to the right of Figures 5.12a and 5.12b. Using Φ as stability measure gives the illusion of increasing stability by large amounts, thus more accuracy is sacrificed (here $\tilde{a}_b \approx 0.885$ for Φ (Figure 5.11) and $\tilde{a}_b \approx 0.930$ for both Φ_{pears} (Figure 5.12a) and for Φ_{iw} (Figure 5.12b)). Optimizing Φ leads to unsatisfactory stability maps with lower predictive performance.

Analogous results to the ones presented in Figures 5.11 and 5.12 for the other data sets are presented in Appendix B. On most data sets, the Pareto-optimal compromises depend on the choice of stability. Even when this is not the case, the choice of measure strongly affects the willingness to sacrifice accuracy, ultimately leading to different chosen Pareto-optimal compromises.

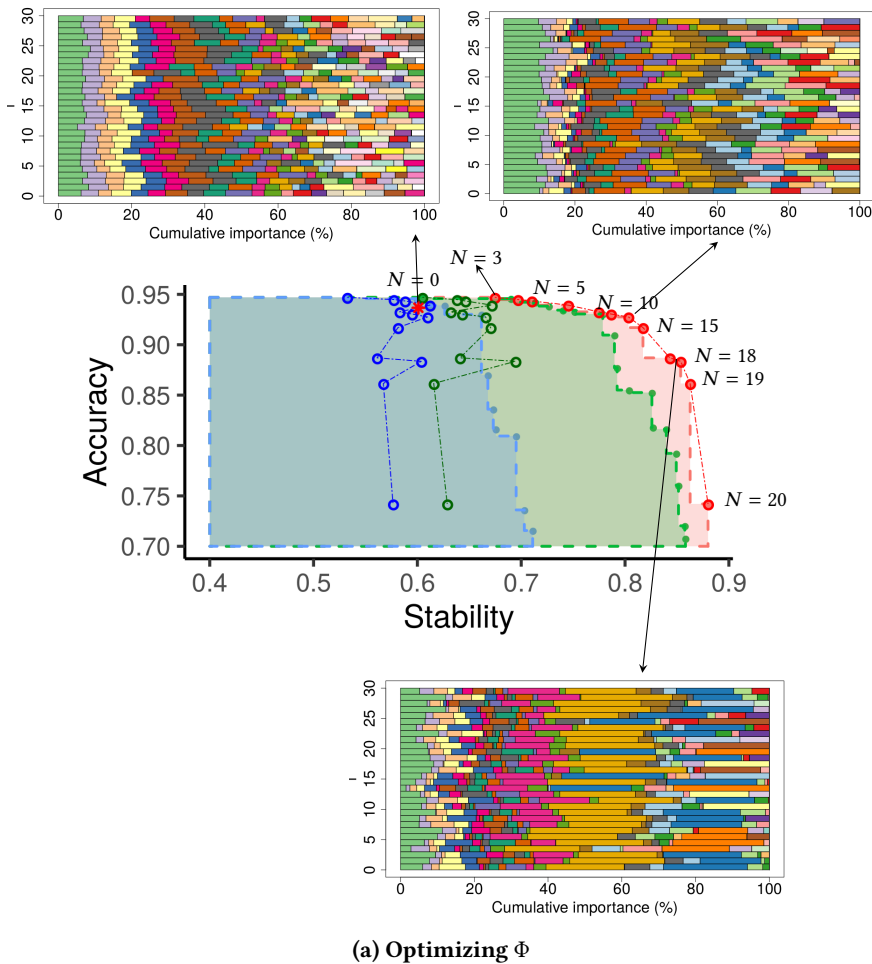


Figure 5.11: Pareto-optimal curves (plain points) for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on the `singh` data set. The circled points are the compromises $(\tilde{a}_b, \Phi)$ (red), $(\tilde{a}_b, \Phi_{\text{pears}})$ (green) and $(\tilde{a}_b, \Phi_{\text{iw}})$ (blue) which are obtained with meta-parameters on the $(\tilde{a}_b, \Phi)$ Pareto-optimal curve. The red star is a typical trade-off obtained with the classic RFE ($N = 0$), which is here slightly off the Pareto-curve. Assuming domain experts have an instance of Equation 3.11 as satisfaction function, the map to the top-right (resp. bottom) is chosen for $0.15 \leq \gamma \leq 0.66$ (resp. $0 \leq \gamma \leq 0.02$). Joint analysis with Figure 5.12 shows that the optimal trade-off found along Pareto-curves in the (accuracy, stability) space clearly depends on the stability measure used and results in strongly different feature stability maps.

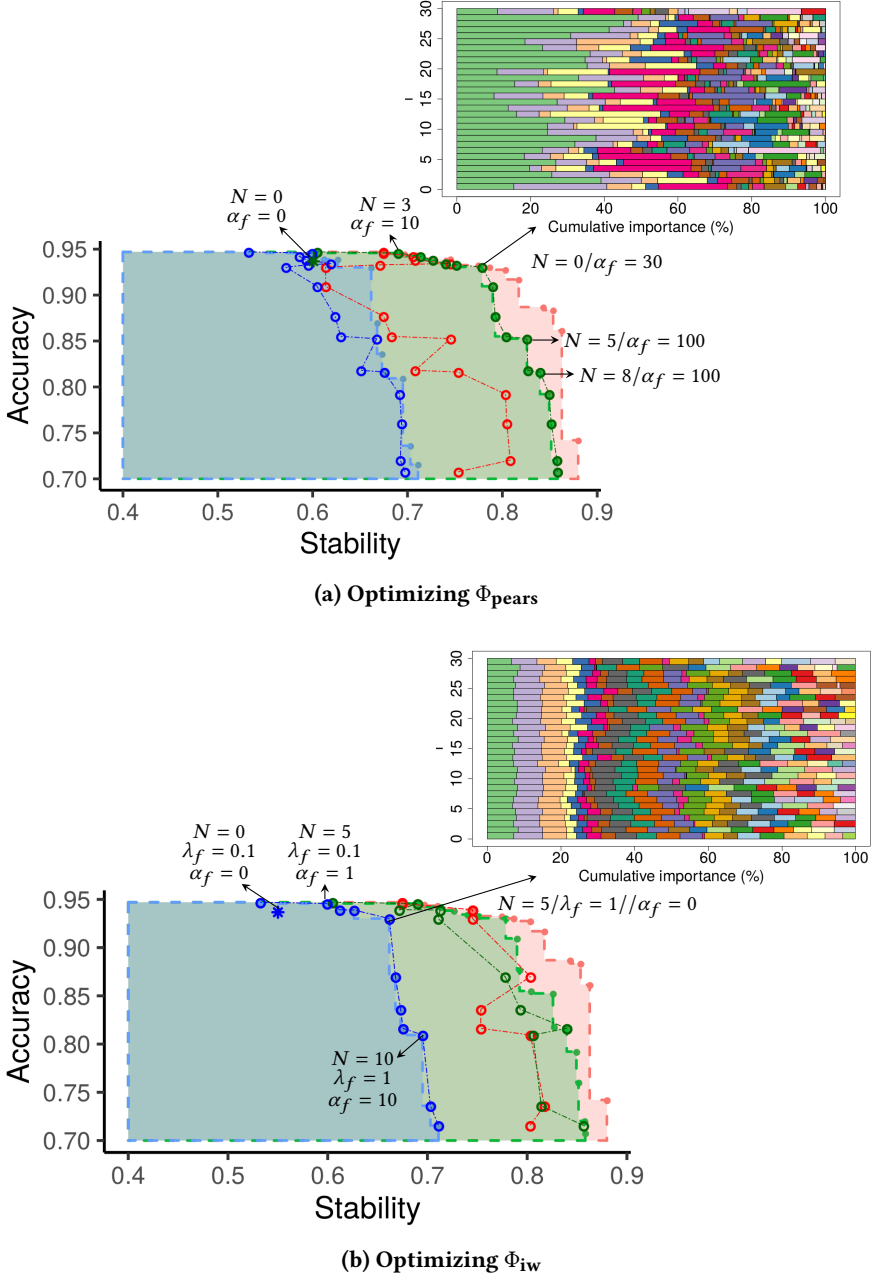


Figure 5.12: Pareto-optimal curves (plain points) for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on the `singh` data set. The circled points are the compromises $(\tilde{a}_b, \Phi)$ (red), $(\tilde{a}_b, \Phi_{\text{pears}})$ (green) and $(\tilde{a}_b, \Phi_{\text{iw}})$ (blue) which are obtained with meta-parameters on the $(\tilde{a}_b, \Phi_{\text{pears}})$ (a) and $(\tilde{a}_b, \Phi_{\text{iw}})$ (b) Pareto-optimal curves. The map to the right of each subfigure is chosen with $0 \leq \gamma \leq 0.75$ (a) and $0 \leq \gamma \leq 0.65$ (b).

5.7 Stability Analysis of Standard Feature Selection Methods

The hybrid-RFE algorithm (Algorithm 3 in Section 4.1) is designed to navigate through the (accuracy, stability) objective space. In Section 5.6, we have shown that using Φ_{iw} as the stability measure improves decision-making in such a context. To broaden our analysis, this section studies the stability of common feature selection methods: logistic regression with the LASSO or ELASTIC NET penalty, random forests, the RELIEF algorithm, and the standard SVM and logistic RFE, which are not designed to explore such a bi-objective space. Results show that our proposed measure Φ_{iw} behaves similarly to Φ and Φ_{pairs} (but still provides some additional insights). This indicates that Φ_{iw} keeps the correct behavior of well-known measures in standard cases (while being more robust to extreme situations, as demonstrated in Section 5.6 and, later, in Section 6.3.2 in the context of feature correlations).

We use additional data sets that are briefly described in Table 5.2. Biomedical data sets (from Table 4.1) are now pruned to 5000 features. We aim at selecting $\min(20, \sqrt{d})$ features, while we set M to 50. We also consider different instance-weighting strategies to increase the impact of the minority class. This can lead to slight (balanced) accuracy gains. We study each selection method independently (aggregated results are provided in Appendix C).

name	year	n	n_{maj}	d	task
ionosphere	1989	350	224	34	radar returns classification
sonar	NA	207	111	60	sonar signals classification
breast	1995	568	357	30	breast cancer diagnosis
gastro	2016	76	55	698	gastrointestinal lesions detect.
lsvt	2014	126	84	310	phonations classification

Table 5.2: Information on the data sets used in this section (in addition to those introduced in Table 4.1), from the UCI machine learning repository.

5.7.1 The LASSO

The LASSO (introduced in Section 2.2.2.1), used in the context of logistic regression, finds the linear model $\mathbf{w}$ minimizing

$$\sum_{i=1}^n \omega_i \log(1 + \exp^{-y_i(\mathbf{w}\mathbf{x}_i)}) + \lambda \|\mathbf{w}\|_1,$$

with ω_i an added per-instance weight. We here consider the strategies where ω_i is set to 1 or to the inverse of the class prior for each example, as evaluated on each bootstrap. The strategy with maximal accuracy is retained. The larger the regularization parameter λ , the fewer features are selected (features with

a non-zero weight). The LASSO is known to be an unstable feature selection approach. Our experiments confirm this (see Appendix C for comparative results). Nonetheless, Figure 5.13 illustrates that the LASSO tends to give more importance (width of the rectangles) to frequently selected features (features at the left of the maps). This positively affects Φ_{iw} . Yet, another kind of instability is often observed with the LASSO: the importance given to each selected feature varies significantly from one selection run to another, which is penalized by Φ_{iw} . This behavior tends to make Φ_{iw} lower than Φ_{pears} as the latter is less sensitive to such an instability. Table 5.3 summarizes the compromises achievable by the LASSO for the three considered stabilities.

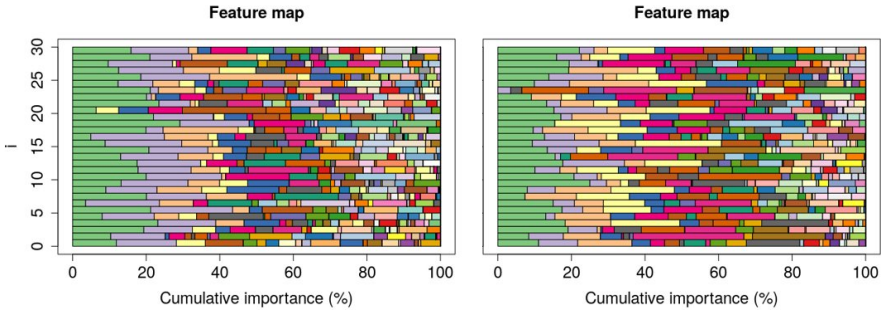


Figure 5.13: Typical feature stability map of the LASSO (on **singh** (left) and **lsvt**(right)). The LASSO regularization parameter is chosen to select $\min(20, \sqrt{d})$ features on average, but the specific number of selected features here varies across runs. The LASSO gives larger importances (width of the rectangles) to frequently selected features (features at the left of the maps). Yet, feature importance is highly varying.

data	A	Φ	Φ_{pears}	Φ_{iw}	data	A	Φ	Φ_{pears}	Φ_{iw}
ion	0.82	0.75	0.89	0.74	son	0.73	0.40	0.49	0.35
bre	0.95	0.69	0.86	0.74	gas	0.79	0.27	0.33	0.23
lsv	0.84	0.38	0.50	0.37	alo	0.78	0.20	0.20	0.15
sin	0.92	0.32	0.59	0.38	chi	0.82	0.28	0.33	0.24
gra	0.72	0.19	0.25	0.17	arc	0.73	0.15	0.23	0.15
bor	0.94	0.17	0.16	0.12					

Table 5.3: Stability of the LASSO on all considered data sets.

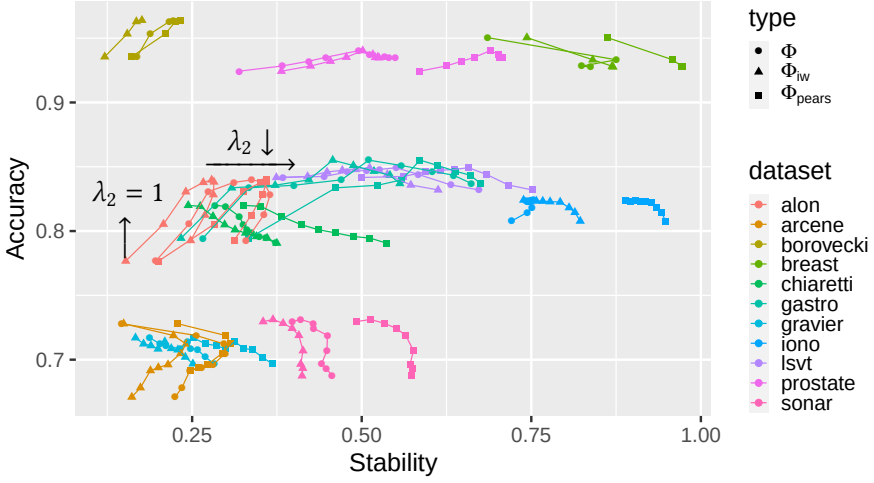


Figure 5.14: Evolution of the (accuracy, stability) compromises obtained with logistic regression with an ELASTIC NET penalty. The reported stability measures are Φ , Φ_{pear} and Φ_{iw} . Each line starts with $\lambda_2 = 1$ (LASSO) which decreases by 0.1 at each point. Stability is generally increased when λ_2 , the weight of the L1 loss in Equation (5.17), decreases. This stability increase sometimes comes at the cost of predictive accuracy.

5.7.2 The Elastic-Net

The ELASTIC NET penalty is a direct generalization to the LASSO penalty which minimizes the linear combination of the L1 and L2 loss

$$\sum_{i=1}^n \omega_i \log(1 + \exp^{-y_i(\mathbf{w}x_i)}) + \lambda_1(\lambda_2 \|\mathbf{w}\|_1 + \frac{(1 - \lambda_2)}{2} \|\mathbf{w}\|_2). \quad (5.17)$$

It is purely equivalent to the LASSO when $\lambda_2 = 1$. Figure 5.14 studies the dependency of the accuracy and stability on the parameter λ_2 . Each line depicts the evolution of a stability measure with (from left to right) a decreasing λ_2 parameter. All three stability measures increase most of the time when departing from the pure LASSO selection ($\lambda_2 = 1$). On some data sets (notably singh, alon, gastro and borovecki), increasing the L2 regularization also increases the accuracy. Then, as λ_2 continues to decrease, the accuracy starts to drop. On other data sets, such as gravier, chiaraetti and breast, the accuracy drops directly when departing from the LASSO selection.

5.7.3 Logistic RFE

The logistic RFE (Algorithm 2 in Section 2.2.2.4) iteratively minimizes Equation (5.17) with $\lambda_2 = 0$ and the least significant features are dropped at each

iteration. After the selection procedure, which uses a given regularization parameter λ , Equation (5.17) is minimized again with a different regularization λ_f for learning the predictive model. Increasing λ_f tends to increase Φ_{iw} and Φ_{pears} as it reduces the instability of the importance of the selected features. As λ_f does not influence the identity of the selected features, Φ remains unchanged with its variations. This phenomenon is illustrated in Figure 5.15 which has been obtained on the `1svt` data set. Increasing the regularization parameter λ used during the selection improves all stability measures. When λ_f is low (left of each line), Φ_{iw} is much lower than Φ due to the instability that occurs during the learning of the final model. Increasing λ_f first increases both the accuracy and Φ_{iw} . The accuracy can then drop when λ_f is increased more. We observed a very similar behavior for the `SVM-RFE` algorithm, even though the latter is generally more unstable (see Appendix C).

5.7.4 Random Forests

Random forests can be used for feature selection as well. Feature importance is computed by randomly permuting the features in the out-of-bag samples of each of the T trees and by computing the predictive accuracy decrease these permutations imply [Bre01b; PVD+13]. The features whose removal causes the largest accuracy decrease are selected, and a new random forest is learned on those features only. We use the `randomForest` R package which allows to oversample the minority class with the `sampsiz` parameter.

Figure 5.16 depicts the (accuracy, stability) trade-offs that are achievable on all data sets. Stability is clearly increased when the forest size grows before converging for $T \approx 1000$. Accuracy also tends to increase, and it converges faster than stability which is consistent with the results of [PVD12]. Figure 5.17 illustrates typical feature stability maps obtained with random forests, here on `gastro`, with $T = 10$ (left) and $T = 3000$ (right). Increasing the size of the forests stabilizes both the selection of the feature subsets and the importance of the selected features. Large forests produce much more interpretable feature stability maps.

5.7.5 The Relief

The `RELIEF` algorithm is a multi-variate filter approach [KR+92]. Feature scores are computed based on feature value differences between neighbor examples. We consider here a version of the `RELIEF` where K nearest instances have equal weights.⁴ The predictive model is a K nearest neighbor classifier with majority voting. This majority voting is weighted by the inverse of the class priors of each neighbor. As can be seen in Figure 5.18, stability is largely

⁴We use the `CORElearn` R package for this purpose.

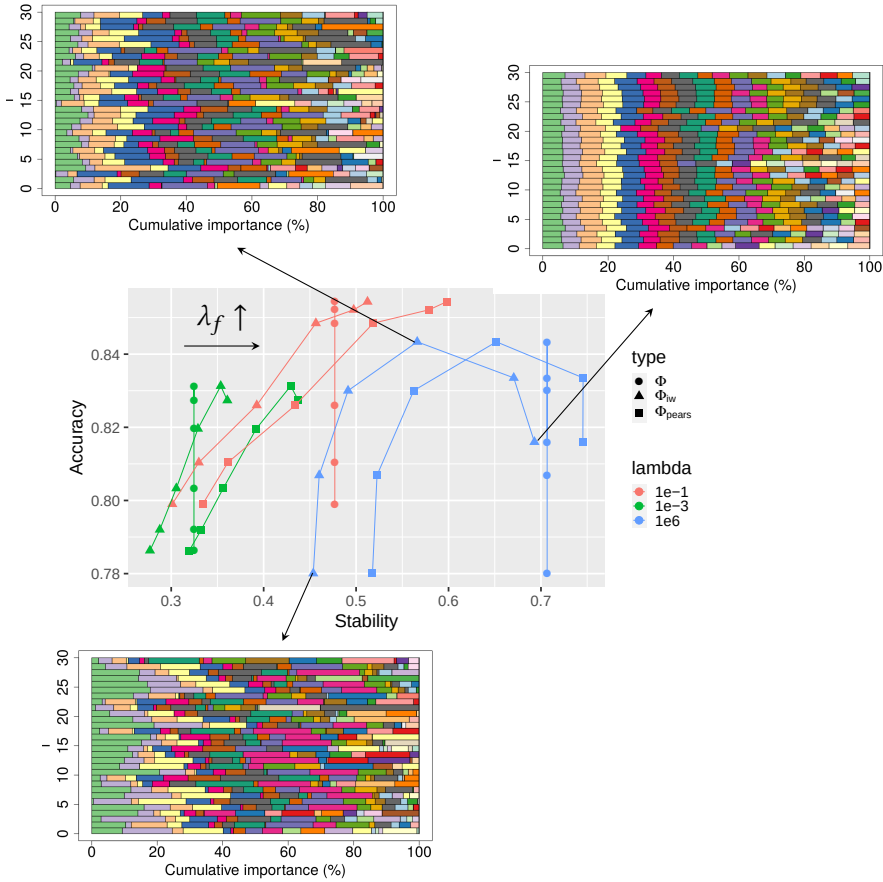


Figure 5.15: Typical results of the logistic RFE (on 1svt). The reported stability measures are Φ , Φ_{pears} and Φ_{iw} . Each line starts (from the left) with a low final regularization parameter λ_f , which is gradually increased. A larger λ_f stabilizes feature importance and improves Φ_{pears} and Φ_{iw} . Increasing the regularization λ improves all three stability measures.

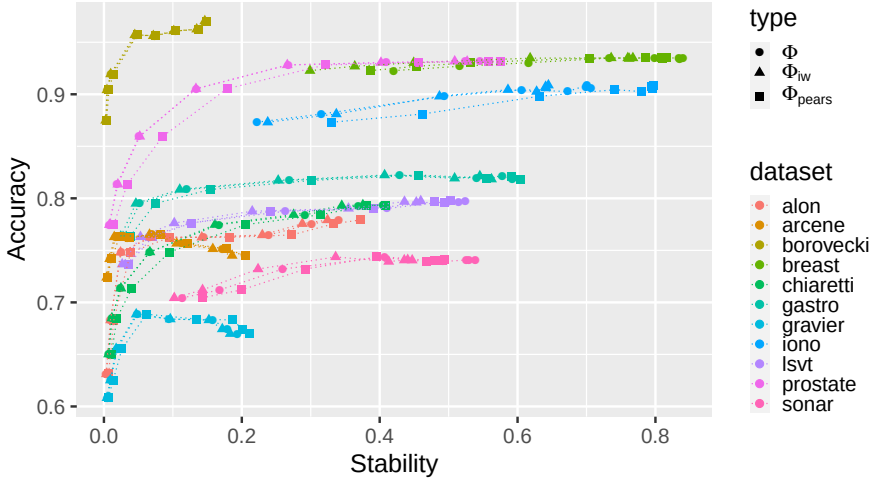


Figure 5.16: (accuracy, stability) compromises reachable with random forests. The reported stability measures are Φ , Φ_{pear} and Φ_{iw} . Each line starts (from the left) with forests of only three trees. The forest size is gradually increased up to 3000 trees. Larger forests are more stable and have a better predictive accuracy.

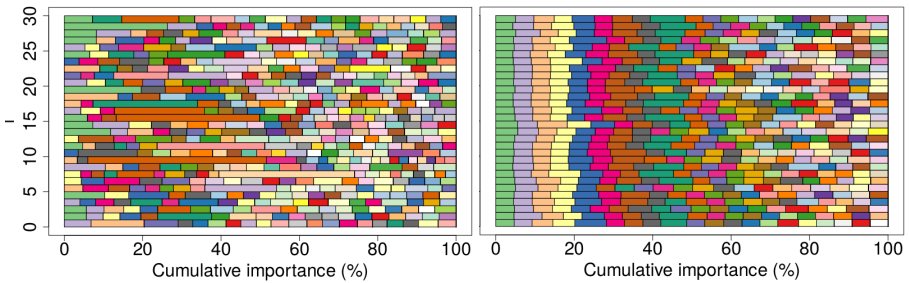


Figure 5.17: Feature stability maps of random forests on **gastro** with $T = 10$ (left) and $T = 3000$ (right). Increasing the size of the random forests produces more interpretable feature stability maps.

influenced (and often increased) by the number K of considered neighbors. Typical features maps obtained with the RELIEF (here on sonar with $K = 1$ (left) and $K = 25$ (right)) are depicted in Figure 5.19. Nearest neighbor algorithms tend to give similar importance to each selected feature.

5.7.6 Deep Feature Selection

Deep Feature Selection is introduced in Section 2.2.2.6. It selects features at the input layer of a neural network by adding a sparse one-to-one layer between the input layer and the first hidden layer. An elastic-net penalty is put on the weights of this one-to-one layer, which creates sparsity.

We use the tensorflow python package [Mar+15] to implement such a family of neural networks. As for all neural networks, this approach has a plethora of meta-parameters. We set some to reasonable values without much consideration while the impact of others on the number of selected features, stability, and predictive performance is evaluated. These meta-parameters are listed below.

- The parameters λ_1^o and λ_2^o of the Elastic-Net penalty put on the one-to-one layer: λ_1^o dictates the overall intensity of the regularization while λ_2^o balances between the L_1 and L_2 losses ($\lambda_2^o = 1 \rightarrow$ Lasso penalty, $\lambda_2^o = 0 \rightarrow$ Ridge penalty).
- Similar to λ_1^o and λ_2^o , λ_1^h and λ_2^h respectively dictate the intensity and smoothness of the Elastic-Net penalty that is put on the weights of the other layers of the neural networks. The same regularization is put on each layer.
- The architecture of the neural network: number of hidden layers and the number of units in each layer. We consider network architectures with up to three hidden layers, of 4, 16, 64 and 256 units. The number of units is kept constant across layers of the same network.
- The batch size is set to 128.
- All weights initializers are set to "random_normal".
- The activation function of the hidden layers is set to the LeakyRelu with the default parameter 0.3 which is defined as

$$\text{LeakyRelu}(x) = \begin{cases} 0.3x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$$

A batch normalization layer is added after each hidden layer.

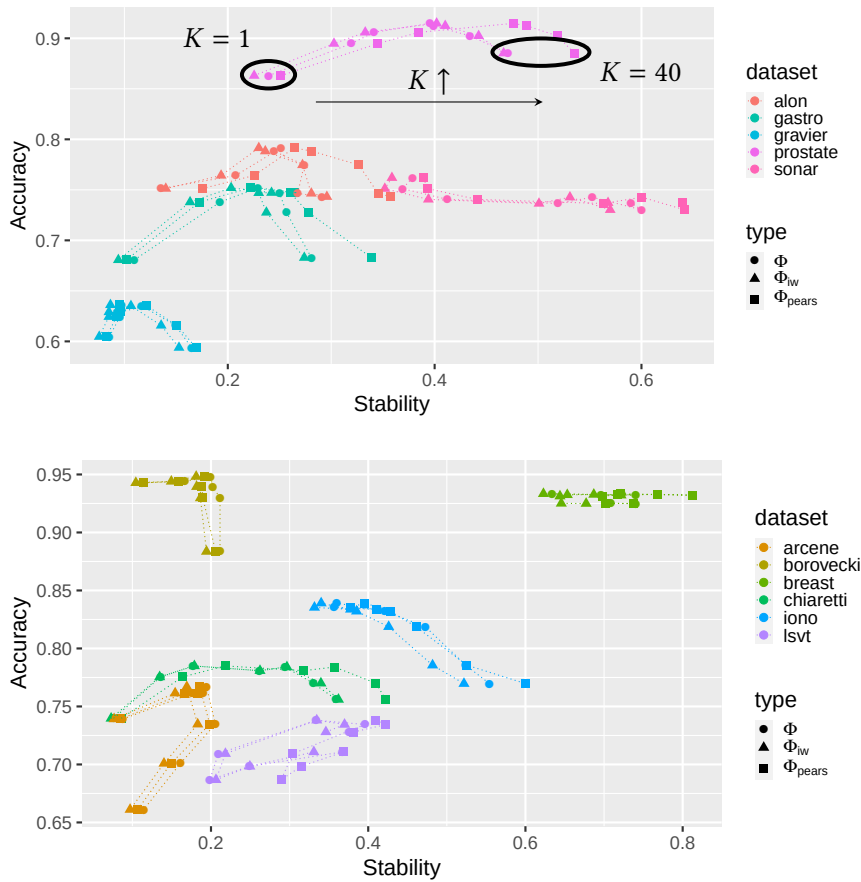


Figure 5.18: (accuracy, stability) compromises of the RELIEF algorithm. The reported stability measures are Φ , Φ_{pears} and Φ_{IW} . Each line starts (from the left) with only one neighbor considered. This number is gradually increased up to 40 which increases stability in most cases.

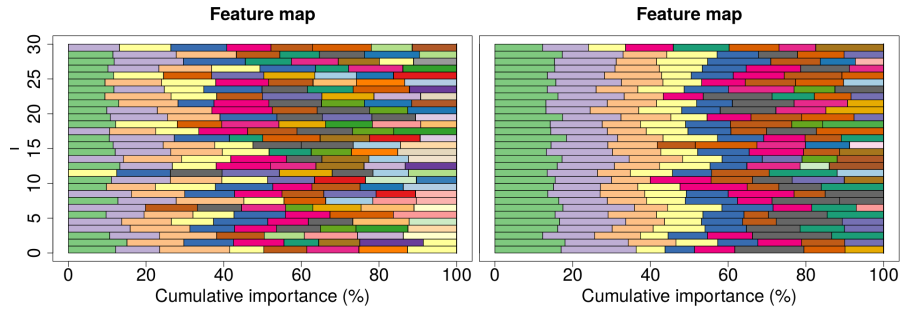


Figure 5.19: Feature stability maps of the RELIEF on sonar with $K = 1$ (left) and $K = 25$ (right).

- The output layer has a softmax activation function. The loss to be minimized by the network is the binary cross-entropy loss.
- We use an exponential-decay learning schedule. We here refer to an iteration as a block of 100 epochs with a constant learning rate. After each iteration, the learning rate is decreased by a factor of 1.5. We consider starting learning rates of 10^{-2} , 10^{-3} and 10^{-4} . Execution is stopped when the number of selected features has not changed in the last three iterations and when the training loss has not decreased more than 2.5% of its value in the last three iterations.

True convergence is rarely reached with neural networks. Consequently, the one-to-one layer weights of the non-selected features never completely reach zero. We adopt the same methodology as in [LCW16] to select features. A feature is considered selected if the absolute value of its weight in the one-to-one layer is higher than one-thousandth of the maximal weight in this layer. Also, as minimizing the loss of a neural network is a non-convex optimization problem, convergence is generally reached at a local minimum. This local minimum could be unsatisfactory (e.g., 1000 features are selected even though we aimed at around 20). Whenever that happens, we automatically restart the learning from scratch. If the same problem arises three times in a row, we discard the set of considered meta-parameters.

Figure 5.20 studies the effect of meta-parameters λ_1^o , λ_2^o , λ_1^h and λ_2^h on the number of selected features on the ionosphere data set. Each point in each subplot corresponds to a meta-parameter combination (while fixing the learning rate to 10^{-3} and using a single hidden layer). Two points are linked if they are obtained by the same meta-parameter combination, except for the studied meta-parameter (λ_1^o (a), λ_2^o (b), λ_1^h (c) and λ_2^h (d)). The intensity of the regularization on the one-to-one layer λ_1^o strongly impacts the number of selected features. The smoothness of this regularization has less impact but still matters. In particular, the number of features selected when there is no L_1 regularization ($\lambda_2^o = 0$) is higher. It is also worth noting that a L_1 regularization on the one-to-one layer is not needed to promote sparsity, as long as there is some amount of L_1 regularization on the hidden layers, and in particular, on the first hidden layer. Whenever this regularization makes all of a neuron's weights null in the first hidden layer, the optimal weight of the corresponding neuron in the one-to-one layer is also null, even with a full L_2 penalty applied to it. Figure 5.20 (c) and (d) further shows that the amount of regularization on the hidden layer λ_1^h also has a strong impact on $\bar{k}$. Similar to λ_2^o , the smoothness of this regularization λ_2^h has less, but still some importance.

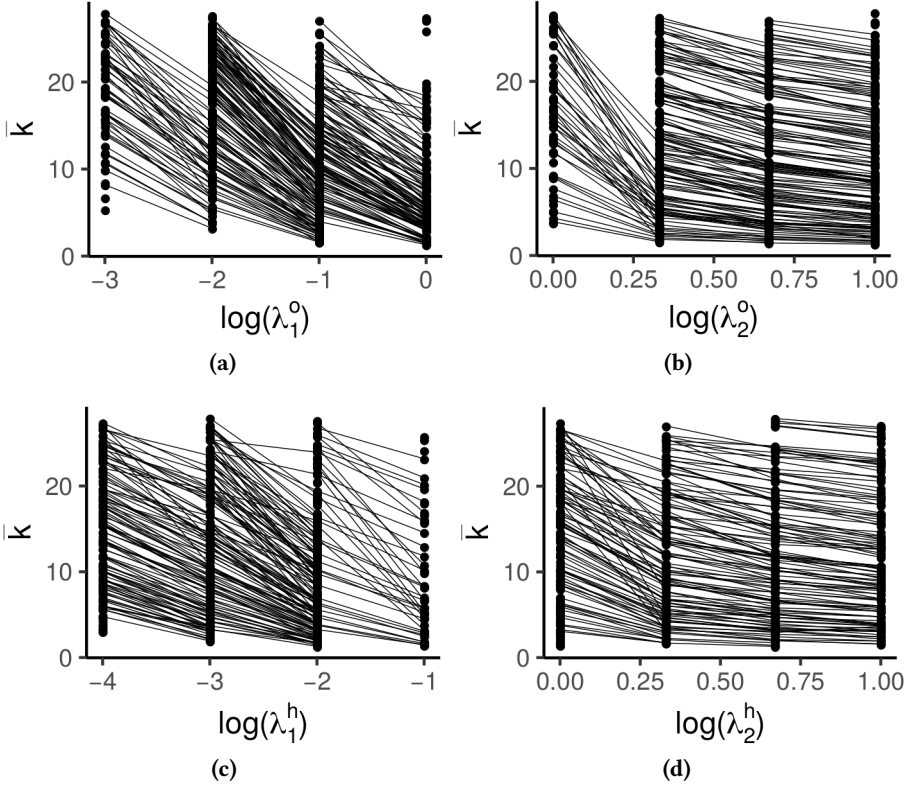


Figure 5.20: Impact of the four regularization parameters on the average number of selected features $\bar{k}$ over the $M = 30$ selection runs. Increasing the amount of regularization (λ_1^o and λ_1^h) decreases $\bar{k}$. Increasing the proportion of L_1 penalty in the regularization also reduces $\bar{k}$.

Figure 5.21 depicts the Pareto-optimal fronts than can be reached for networks with up to three hidden layers, on 4 data sets: ionosphere, sonar, breast and 1svt. The number of units of each hidden layer is constant and set to 256.⁵ The network without any hidden layer (which learns linear models) consistently reaches the highest stability. On ionosphere and sonar, adding a hidden layer improves the accuracy but at the cost of selection stability. Adding more layers continues to reduce selection stability without any gain in classification performance. On breast, there is no significant gain in classification performance by adding hidden layers. Yet, stability is still decreased. On 1svt, the network without any hidden layer outperforms the others in terms of classification performance and selection stability. Such behavior is common across the microarray data sets (see Appendix C) as they have low sample-to-feature ratios, and deep neural networks are known to

⁵No strong dependency on the number of units is observed, as studied later in Figure 5.22

need large amounts of data to work properly.

The effect of the number of units in the hidden layers is studied in Figure 5.22. The impact of increasing the number of hidden units (kept constant in all hidden layers) seems less clear and more data set dependent.

Figure 5.23 depicts two typical feature maps obtained with Deep Feature Selection on ionosphere. The left one is obtained with a network of a single hidden layer of 256 units. The right one is achieved with a network without any hidden layer. The interpretability of the right feature map is clearly better, but it may come at the cost of some predictive performance (here $\approx 4\%$).

In the comparative results of Appendix C, we compare the three architectures studied in Figure 5.21: a neural network without any hidden layer (shallow-dfs), a neural network with a single hidden layer of 256 hidden units (dfs), and a neural network with three layers of 256 units each (deep-dfs). Shallow-dfs tends to have a similar classification performance as the other linear methods (elastic-net, svm and logistic-RFE) but with a consistently reduced selection stability. Dfs and deep-dfs sometimes improve the classification performance of shallow-dfs, but that comes at a high cost of selection stability. Deep-dfs only outperforms dfs and shallow dfs on arcene (in terms of balanced accuracy).

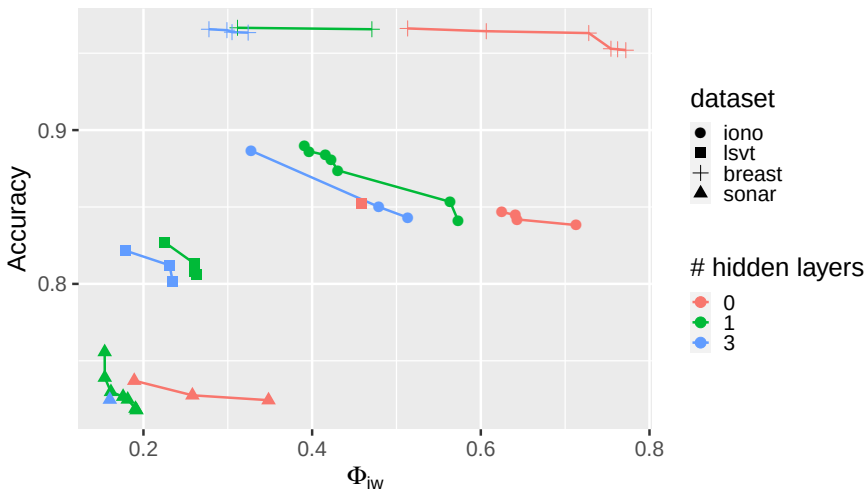


Figure 5.21: Pareto fronts with networks of 0, 1, and 3 hidden layers of 256 units. Linear networks without hidden layers are the most stable but sometimes have a sub-optimal predictive performance. Adding layers consistently deteriorates selection stability.

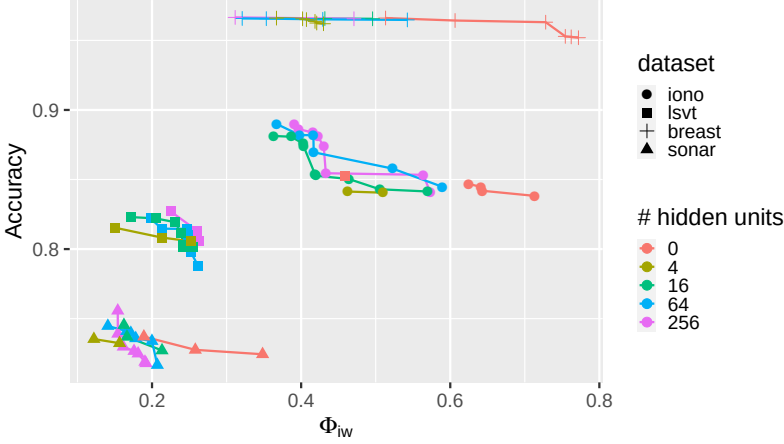


Figure 5.22: Pareto-fronts of neural networks with 4, 16, 64 and 256 units in their unique hidden layer. The influence of the number of units is less clear.

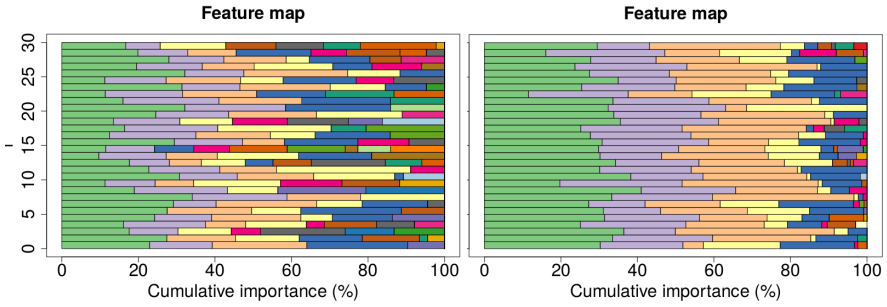


Figure 5.23: Feature stability maps of Deep Feature Selection on *ionosphere* with a hidden layer of 16 neurons (left) and without any hidden layer (right). Both selects ≈ 6.25 features on average. The network without hidden layers gives more importance to frequently selected features.

5.8 Variability of Stability

Taking into account the importance of selected features in the stability value creates an additional degree of variability that could result in unreliable stability estimation when the number of selection runs M is small. We show here that this added variability is usually small. Furthermore, the variability of our measure Φ_{iw} is generally lower than the variability of Φ_{pears} . Figure 5.24 depicts the variation coefficients $c_v(\triangleq \frac{\sigma}{\mu})$ of Φ , Φ_{pears} and Φ_{iw} , when 200 stability estimates are measured using $M = 5$ with the LASSO. As the LASSO exhibits the largest feature importance variability, it best highlights the differences between stability measures. On 6 of the 11 data sets, $c_{v,\Phi} < c_{v,\Phi_{iw}} < c_{v,\Phi_{\text{pears}}}$,

there is no noticeable difference on 4, and on ionosphere only, $c_{v,\Phi_{\text{pears}}} < c_{v,\Phi_{\text{iw}}} < c_{v,\Phi}$. Using importance-based measures indeed creates an additional uncertainty in the stability value, but our proposed measure Φ_{iw} is, in general, less impacted than Φ_{pears} .

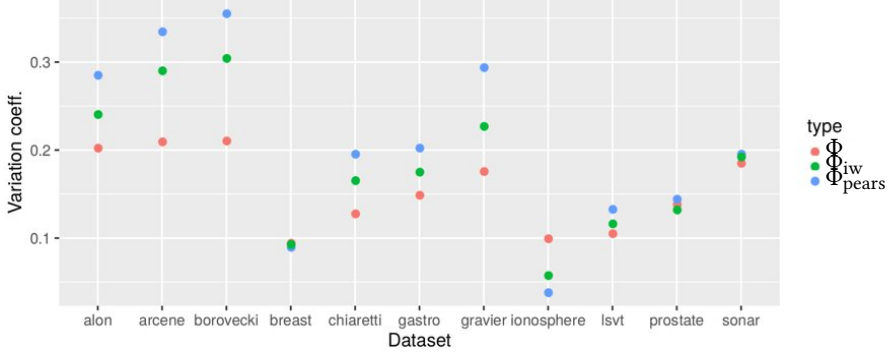


Figure 5.24: Variation coefficients of the stability measures obtained with the LASSO regression on $M = 1000$ selection runs. Stability estimates are found by aggregating groups of 5 runs (thus producing 200 of such estimates). In most cases, $c_{v,\Phi} < c_{v,\Phi_{\text{iw}}} < c_{v,\Phi_{\text{pears}}}$.

5.9 Conclusion

In the context of stability optimization, adequately measuring stability is necessary for domain experts to make informed decisions. While most measures proposed in the literature study the stability of feature subsets, we incorporate the relative importance of the selected features in predictive models into the stability value. In particular, we propose a generic way to evaluate the importance of each selected feature in a predictive model and consider the variability of this importance as an additional source of instability.

Our proposed measure Φ_{iw} is proven to be fully defined, bounded, corrected for chance, and maximal if and only if the importance distribution is deterministic. Confidence intervals and hypothesis tests on the measure's asymptotic value are built and validated on real data. Furthermore, it is shown to improve decision-making when stability is jointly optimized with predictive accuracy. We demonstrate this phenomenon theoretically under the form of a new property, the *importance weighted decomposition* property, that only Φ_{iw} satisfies. We further show that using current measures in such a bi-objective context can lead to unsatisfactory results and often gives a false sense of stability which hinders appropriate decision-making. This is illustrated experimentally on microarray and mass-spectrometric data.

We also propose a simple visualization tool, referred to as feature stability maps, which allows the intuitive estimation of stability across several runs of

feature selection. Inspection of such feature stability maps allows the identification of the most reliably identified features along with the most important features in the predictive models. Features combining both properties are likely to be particularly appealing to domain experts.

Finally, we demonstrate on multiple data sources that our novel measure Φ_{iw} keeps the correct behavior of well-known measures in standard cases (while being more robust for stability optimization). Still, we use it to provide new insights into the stability of several standard feature selection methods, such as logistic regression with an elastic-net penalty or random forests.

Stability in Correlated Spaces

6

This Chapter deals with stability estimation in the presence of feature correlations. Section 6.1 introduces the problem and its associated challenges. Section 6.2 reviews existing stability measures developed specifically to deal with such correlations. Then, Section 6.3 shows the limitations of these measures in two toy scenarios and using simulations. While the importance weighted stability Φ_{iw} does not integrate feature correlations, this section shows that it can still improve the behavior of current measures when the feature space is composed of highly correlated feature groups. This is also theoretically justified under the form of a property. Nevertheless, Φ_{iw} remains generally unsatisfactory when dealing with feature correlations. A new measure, Φ_{msi} , is then proposed in Section 6.4. It is shown to be more adequate in all the scenarios studied in Section 6.3. Section 6.5 then evaluates the stability of standard feature selection methods according to our new index and shows that the (accuracy, stability) Pareto-optimal fronts change when feature correlation is taken into account. This emphasizes the need to use our correlation-aware measure Φ_{msi} . This chapter corresponds to the work published in [HD21a; HD21b].

6.1 Problem Statement

This section introduces the issue of measuring stability in the presence of feature correlations. In this Chapter, we use the keywords *feature correlation*, *feature similarity* and *feature redundancy* interchangeably.

Inspired by Sechidis et al. [Sec+19], we first consider a scenario where a selection algorithm toggles between pairs of correlated variables across runs. More specifically, let us represent the selection matrix $\mathcal{Z}$ below with one row per selection run, $z_{i,f}$ indicating the selection of feature f in run i

$$\mathcal{Z} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & \dots & 0 \end{pmatrix} \begin{array}{l} \mathbf{z}_{1,-}, \text{ selected features in run 1} \\ \mathbf{z}_{2,-}, \text{ selected features in run 2} \\ \dots \end{array} \quad (6.1)$$

In this limit case scenario, the selection algorithm toggles between fea-

tures (1,2) and between features (3,4), while no other feature is ever selected. Let us assume that each pair of these features is strongly correlated. Since the selection algorithm always picks one variable in each correlated group, the information captured is essentially the same in each run, and the selection should be considered perfectly stable. So far, all the stability measures studied in this thesis are only functions of the identity of the selected features and sometimes of their importance in predictive models. In other words, possible similarities between features are plainly ignored. For the example above, Φ and Φ_{iw} (assuming a feature importance of 1 for all selected features) would tend to $\frac{1}{3}$ as d tends to infinity. The desired stability value is 1 however. Ignoring feature similarities in the stability computation systematically offers a pessimistic view of stability. A feature selection algorithm selecting different feature sets in each run is seen as unstable, even if the features in these sets are highly similar to each other. This is particularly problematic in the biomedical domain, where the input features (e.g., gene expressions) often display strong correlations.

Estimating stability in the presence of feature correlation is another prime example where taking into account the importance of the selected features in predictive models is necessary. Yet, to the best of our knowledge, no existing stability measure depends on both feature importance and feature correlation. The following example illustrates such a necessity. In the first selection run, features f_1 and f_2 are selected and the predictive model is simply the sign of their sum

$$y = \text{sign}(f_1 + f_2).$$

In the second run, features f_2 , f_3 and f_4 are selected and the predictive model is

$$y = \text{sign}(f_2 + 0.5f_3 + 0.5f_4).$$

For such linear models, the importance of the selected features are simply the absolute value of their weight in the model. Features f_1 and f_2 have an importance of 1 in the first run, while features f_2 , f_3 and f_4 have an importance of 1, 0.5 and 0.5 in the second run, respectively. The importance matrix I is then

$$I = \frac{\bar{k}}{2} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0.5 & 0.5 & 0 & 0 & \dots & 0 \end{pmatrix}. \quad (6.2)$$

At first sight, the algorithm seems somewhat stable, but not completely. Feature f_2 is selected both times with the same importance, but the other features differ. Still, let us assume that f_1 is in fact highly correlated to both f_3 and f_4 . In that case, taking $0.5f_3 + 0.5f_4$ or just f_1 does not matter. Both predictive models are equivalent, and stability should be maximal. There is a clear implicit matching between f_1 in the first run and both f_3 and f_4 in the second run, and between f_2 in both runs. The part of the predictive model using f_1

in the first run is equivalent to the part of the predictive model using f_3 and f_4 in the second run, while the parts of both predictive models using f_2 are also equivalent. One could say that f_1 in run 1 matches its importance with f_3 and f_4 in run 2, and that f_2 in run 1 matches its importance with itself in run 2. The measure Φ_{msi} , which relies more formally on this notion of matched importance, is proposed in this chapter. Ignoring feature importance does not allow us to deal correctly with these kinds of scenarios. A critical part of the reasoning here lies in the fact that the importance of f_1 in run 1 is equal to the sum of the importances of f_3 and f_4 in run 2: $I_{1,f_1} = I_{2,f_3} + I_{2,f_4}$.

More formally, we note the similarity between feature f and feature g as $s_{f,g}$. This quantity is bounded in $[0, 1]$. The minimum value of 0 indicates no similarity between these features, while the maximum value of 1 specifies a perfect similarity. The similarity matrix is noted S , and has the $s_{f,g}$ values as elements. It is supposed to be *a priori* defined or estimated over the whole training set for any pair of features. Typical choices include the absolute values of Pearson's or Spearman's correlation coefficients, or mutual information normalized in the $[0, 1]$ interval over the whole training set. Such strategies would capture feature similarity as explained by the data. The matrix S could also be constructed from domain knowledge, thus representing feature similarity as explained by domain experts. The only constraints on S for stability to be well-defined are that it must be symmetrical, the main diagonal entries must all be equal to 1, and the off-diagonal elements must be between 0 and 1. The challenge studied in this chapter consists in finding a proper way to use the similarity matrix S to compute the stability of feature selection over M runs.

6.2 Current Stability Measures

This section reviews existing stability measures that are specifically designed to deal with feature correlations. Their limitations are then studied in Section 6.3.

6.2.1 Sechidis Index

Sechidis et al. [Sec+19] generalize the index Φ to accurately measure selection stability in the presence of high correlations between variables. The idea behind the measure is that an algorithm that tends to select different features should not be considered unstable if these features are highly correlated to each other, as the *effective* extracted information is the same. To this goal, they define the *effective* similarity between two selection runs as the generalized inner product

$$|\mathcal{F}_i \cap \mathcal{F}_j|_S = \mathbf{z}_i^T S \mathbf{z}_j \quad (6.3)$$

where $z_{i,f}$ is the Bernoulli variable which is equal to 1 when feature f is selected in selection run i and where the elements $s_{f,f'} \geq 0$ of the matrix $\mathcal{S}$ represent the correlation between feature f and f' . The more correlated the selected variables in run i and j are to each other, the bigger the similarity between these runs. They then proved the following result:

$$\frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\mathcal{S}} = \bar{k}_{\mathcal{S}} - \text{tr}(\mathcal{S}K_{\mathcal{Z}\mathcal{Z}}) \quad (6.4)$$

where $\bar{k}_{\mathcal{S}} = \frac{1}{M} \sum_{i=1}^M \mathbf{z}_i \mathcal{S} \mathbf{z}_i$ and with $K_{\mathcal{Z}\mathcal{Z}}$ the variance-covariance matrix of $\mathcal{Z}$, the matrix with the elements $z_{i,f}$. Equation (6.4) is analogous to Equation (2.38) when $\mathcal{S}$ is the identity matrix. The following frequency-based measure can then be derived

$$\Phi_{\mathcal{S}} = 1 - \frac{\text{tr}(\mathcal{S}K_{\mathcal{Z}\mathcal{Z}})}{\text{tr}(\mathcal{S}\Sigma^0)} \quad (6.5)$$

with Σ^0 the matrix that normalizes the measure. This matrix has $\Sigma_{f,f}^0 = \frac{\bar{k}}{d}(1 - \frac{\bar{k}}{d})$ as main diagonal elements, and $\Sigma_{f,g}^0 = \frac{\bar{k}^2 - \bar{k}}{d^2 - d} - \frac{\bar{k}^2}{d^2}$ as off-diagonal elements. This stability index $\Phi_{\mathcal{S}}$ reduces to Φ whenever $\mathcal{S}$ is the identity matrix. For further details, we refer the reader to [Sec+19].

In the limit case scenario presented above (equation 6.1) and assuming a perfect correlation between features 1 and 2, and, 3 and 4, respectively, $\Phi_{\mathcal{S}} = 1$. Yet, we show in section 6.3 that this measure is not lower-bounded and does not behave appropriately in some cases. Section 6.6 details the issue of the lower-bound in more detail and shows why it can tend to $-\infty$ even in seemingly stable situations.

6.2.2 POGR Index

The POGR index is another existing stability measure defined to handle feature correlations [Zha+09]:

$$\text{POGR} = \frac{1}{M(M-1)} \sum_{i=1}^M \sum_{j=1, j \neq i}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j| + O_{i,j}}{k_i}, \quad (6.6)$$

with $|\mathcal{F}_i \cap \mathcal{F}_j|$ the number of features selected in both runs i and j , and $O_{i,j}$ the number of selected features in run i which are not selected in run j but are significantly correlated to at least one feature selected in run j . Two features f and g are significantly correlated whenever their similarity $s_{f,g}$ is above a given threshold δ . In our limit case scenario (6.1), $\text{POGR}=1$, as the algorithm always selects a feature from each of the two correlated pairs. Yet, we also illustrate its limitations in Section 6.3.

6.2.3 Yu Index: bipartite matching

Similarly to our proposal (detailed in section 6.4), Yu et al. [YDL08] define a stability measure as the objective value of a maximum weighted bipartite matching problem. The two sets of vertices represent the selected features of two selection runs, while the edge weights are the correlations between these features. The authors also propose a variant of their measure where each vertex can represent a correlated feature group as a whole. This measure is, however, not *fully defined*, as it requires the number of vertices (individually selected features or feature groups) to be constant across selection runs. This restriction is hardly met by all selection algorithms that could be considered in practice. In contrast, we show in section 6.4 that our novel measure is fully defined and generalizes the measure proposed in [YDL08].

6.3 Limitations of current measures

This section demonstrates the limitations of existing stability measures when the feature space is composed of highly correlated feature groups. Section 6.3.1 illustrates this on two toy scenarios while Section 6.3.2 performs simulations. The importance-weighted measure is shown to improve the behavior of current measures in this context, as is further justified theoretically by a property that only Φ_{iw} satisfies. Nonetheless, Φ_{iw} remains generally unsatisfactory as it does not explicitly handle feature correlation. A novel measure, Φ_{msi} , is then proposed in Section 6.4.

6.3.1 Toy scenarios

We describe here two scenarios of feature selection in correlated feature spaces. We show that existing stability measures exhibit undesirable behaviors in these scenarios, which motivates our novel stability measure introduced in section 6.4.

Figures 6.1 and 6.2 illustrate the similarity between two selection runs. Like for the feature stability maps introduced in Section 5.1, each feature (or here each group of highly correlated features) is identified by a unique color, and the width of a rectangle corresponds to the relative importance value of the feature (group) in the predictive model of this run. We assume, for simplicity, that the total number d of features tends to infinity, and that the only non-negligible feature correlations are the ones illustrated by explicit links in the figures. This implies that the denominator of Φ_S in equation (6.5) simply becomes $\text{tr}(\mathbf{S}\Sigma^0) = \bar{k}$, the average number of selected features.

In both scenarios, we consider a group of q perfectly correlated features, and we study what happens when q , starting from 1, gradually increases. The importance values of the features inside a correlated group can either be

concentrated on a single feature in the group or, to the contrary, be divided, possibly unevenly, among several or all correlated features. No matter the case, we assume that the cumulative importance of all features within a correlated group roughly stays constant across selection runs. This assumption is actually confirmed in our practical experiments reported in Figure 6.3 in Section 6.3.2. Our limit case scenarios are designed in such a way that the stability value should be equal to $\frac{1}{2}$ and stay constant no matter the value of q . We show that no existing stability index satisfies this property.

In the first scenario (Figure 6.1), a group of q perfectly correlated features (orange) is selected with a *cumulative* importance of $\frac{k}{4}$ in both selection runs, together with another feature (green). The two additional selected features differ in each run. Arguably, whether a large group of correlated features or a single feature from this group is selected should not impact stability as long as the global contribution of the group to the predictive models is unchanged. In such a scenario, we argue that the stability should be equal to $\frac{1}{2}$, independently of q , as half of the selected information is common between both runs. In this scenario, Φ_S , Φ , Φ_{pairs} , Φ_{iw} and POGR correctly start at $\frac{1}{2}$ when $q = 1$ but Φ_S , Φ (both are equivalent in this example), and POGR increase and tend to 1 when q tends to ∞ while Φ_{pairs} decreases with q . As for the importance weighted measure Φ_{iw} , it remains equal to the correct value of $\frac{1}{2}$ under the assumption that importance is distributed evenly¹ among the q correlated features. Section 6.3.2 illustrates an example where this assumption holds true (with the Group LASSO), but this need not be the case in general. Under no assumption, Φ_{iw} is upper-bounded by $\frac{1}{2}$, its exact value depending on the importance distribution of each variable in the correlated group. The same assumption is made to compute the closed formula for Φ_{pairs} , but the limit case is exact provided that the importance of each correlated variable tends to 0 when q tends to infinity. The precise dependencies of these measures on q are summarized in Table 6.1.

The second scenario (Figure 6.2) is nearly identical to the first one, except that a single feature of the correlated group is selected in the first selection run. This feature captures the whole group's importance. Again, stability should be equal to $\frac{1}{2}$, independently of q . In this second scenario, Φ_S , Φ , Φ_{pairs} , Φ_{iw} and POGR correctly start at $\frac{1}{2}$ when $q = 1$ but Φ_S , Φ , Φ_{pairs} and Φ_{iw} decrease and respectively tend to $-\infty$, 0, $\frac{\sqrt{3}}{6}$ and $\frac{1}{4}$ when $q \rightarrow \infty$, while POGR increases and tends to $\frac{3}{4}$. Closed formulas for Φ_{pairs} and Φ_{iw} are again obtained under the assumption of an even importance distribution inside the correlated group. Yet, the limit cases are both exact whenever the importance

¹Actually, as detailed in Section 6.3.2, the assumption that importance is distributed *deterministically* among the q features is sufficient (*i.e* each feature of the group has the same importance in both runs). The assumption is lighter than an *even* distribution.



Figure 6.1: Scenario 1. A group of q perfectly correlated features (orange) is selected in both selection runs. Their specific importance in predictive models may vary but their *cumulative* importance is assumed constant across runs and equal to $\frac{\bar{k}}{4}$. The feature in green is also selected in both runs with a constant importance. The 2 additional features differ in each run. The stability should be equal to $\frac{1}{2}$ in such a scenario, independently of the q value, because the information captured by the left half of the predictive models is essentially the same in both runs.

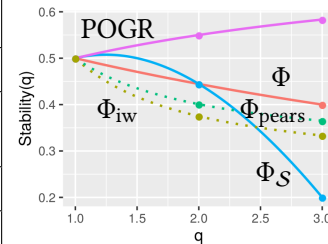
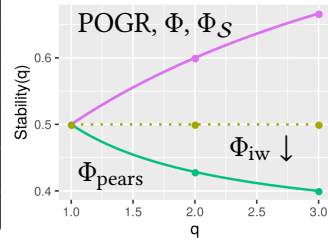


Figure 6.2: Scenario 2. In the second selection run (below), a group of q perfectly correlated features is selected with a cumulative importance of $\frac{\bar{k}}{4}$. In the first run (top), a single feature from this group is selected and captures the whole group importance in the predictive model. The stability should be equal to $\frac{1}{2}$, independently of the q value.

Table 6.1: Dependencies of Φ_S , Φ , Φ_{pears} and POGR on the size q of the correlated group in the scenarios represented in Figures 6.1 (top) and 6.2 (bottom). In order to get a closed formula for Φ_{pears} and Φ_{iw} , importance values are assumed to be evenly distributed within a correlated group.

Measure	Value	$q = 1$	$\lim_{q \rightarrow \infty}$
Φ_S	$\frac{q+1}{q+3}$	$\frac{1}{2}$	1
Φ	$\frac{q+1}{q+3}$	$\frac{1}{2}$	1
Φ_{pears}	$\approx \frac{q+1}{3q+1}$	$\frac{1}{2}$	$\frac{1}{3}$
POGR	$\frac{q+1}{q+3}$	$\frac{1}{2}$	1
Φ_{iw}	$\leq \frac{1}{2}$	$\frac{1}{2}$	$\leq \frac{1}{2}$

Measure	Value	$q = 1$	$\lim_{q \rightarrow \infty}$
Φ_S	$1 - \frac{q^2 - 2q + 5}{q+7}$	$\frac{1}{2}$	$-\infty$
Φ	$1 - \frac{q+3}{q+7}$	$\frac{1}{2}$	0
Φ_{pears}	$\approx \frac{\frac{1}{4q} + \frac{1}{4}}{\sqrt{\frac{1}{4q} + \frac{3}{4}}}$	$\frac{1}{2}$	$\frac{\sqrt{3}}{6}$
POGR	$\frac{1}{2} \left(\frac{1}{2} + \frac{q+1}{q+3} \right)$	$\frac{1}{2}$	$\frac{3}{4}$
Φ_{iw}	$\approx \frac{q+1}{4q}$	$\frac{1}{2}$	$\frac{1}{4}$



of each correlated variable tends to 0 when q tends to infinity.

To conclude this section, no existing stability measure is satisfactory for either of these two toy scenarios. The importance-weighted measure is the only one to behave adequately in the first scenario, but only under an assumption that needs not always be true. It can do so as it deals appropriately with feature importance, unlike the other measures. In the next section, we further illustrate these limitations in a simulated setting.

6.3.2 Simulation Studies I

We use an artificially generated data set with $N = 5$ groups of variables. Each group contains q features that are highly correlated to each other (average correlation of $\rho^g \gg 0$). In addition to these feature groups, the data set contains $l = 1000$ variables. Feature values are sampled from two multivariate normal distributions using the `mvrnorm` R package. Positive examples ($n_+ = 100$) are sampled from a first distribution, centered on μ_+ , a vector with $\mu_{+,f} = \mu_+^g$ if feature f belongs to one of the $N = 5$ correlated groups, μ_+^{-g} otherwise. Negative examples ($n_- = 100$) are sampled from a second distribution, centered on $\mu_- = -\mu_+$. Both distributions have unit variance. We consider three scenarios with different values of μ_+^g , μ_+^{-g} and ρ^g , specified in Table 6.2. In all scenarios, features inside a correlated group are very relevant to the binary prediction task, while features outside such groups are less but still marginally relevant. For feature selection, we use the group LASSO (scenarios 1 and 2), configured such that it selects all features inside a group or none of them, and the standard LASSO (scenario 3). We set the regularization parameter λ of the LASSO and group LASSO such as to select approximately 40 features when the size q of the correlated groups is equal to 1. The $N = 5$ feature groups are thus expected to be selected in most of the $M = 30$ selection runs, while the selection of the additional features should be unstable. The experiment is repeated ten times using different generative seeds for the data sets, and the mean stability values are reported in Figure 6.4 as a function of q , the size of the correlated groups.

We first study Figure 6.3 which represents the cumulative importance of the features that are selected by the group LASSO in scenario 1 when $q = 1$ (top) and $q = 10$ (bottom). Clearly, the group LASSO gives more importance to the features of the five groups when $q = 1$ as they are more relevant (by design). When $q = 10$, however, the importance of the features inside the correlated groups is reduced, such that the cumulative importance of each group is approximately the same as in the $q = 1$ case. In the same spirit as in [Sec+19] where the authors argue that the alternate selection of highly correlated features should not influence stability, we argue that the size of the correlated groups q should not drastically influence stability either (whenever

Scenario	μ_+^g	μ_+^{-g}	ρ^g	method
1	0.35	0.05	0.8	group LASSO
2	0.5	0.05	0.8	group LASSO
3	0.5	0.05	0.95	LASSO

Table 6.2: Experimental settings for the three studied scenarios. The relevance of features inside one of the $N = 5$ correlated groups is related to μ_+^g while the relevance of features outside any group ($\sim \mu_+^{-g}$) is constant across scenarios. The average intra-group correlation is ρ^g and inter-group correlation is negligible. In scenarios 2 and 3, features inside correlated groups are very relevant and are selected in (nearly) all runs.

the group importance is independent of q), as the *effective* extracted information is unchanged.

In this controlled experiment, the similarity matrix $\mathcal{S}$ is estimated as the absolute values of the pairwise Spearman's ρ correlation: $s_{f,f'} = |\rho_{f,f'}|$. Such similarities are used when computing Φ_{msi} and $\Phi_{\mathcal{S}}$. Figure 6.4 compares Φ , Φ_{iw} , Φ_{pears} (with the feature weights $w_{f,i}$ as the importances $I_{i,f}$), $\Phi_{\mathcal{S}}$ and POGR when the sizes of the correlated groups c vary. To compute POGR, two features are assumed to be significantly correlated if and only if they belong to the same correlated group.

When the group LASSO is used (Figures 6.4a and 6.4b), the standard stability Φ increases when the number of correlated variables inside each group grows, which is undesirable. Increasing the number of correlated variables increases Φ because their small selection variance is counted more than once. With q_g , the size of the correlated group g , p_g its selection frequency and $s_g^2 = p_g(1 - p_g)$, its selection variance (we assume here $M \rightarrow \infty$ to simplify calculations),

$$\Phi = 1 - \frac{\sum_g \sum_{f \in g} s_g^2}{\sum_g p_g q_g} \triangleq 1 - \frac{\sum_g q_g \times s_g^2}{\sum_g p_g q_g}. \quad (6.7)$$

In Equation (6.7), a variable outside any correlated group is considered as being in a group of size 1. In the above scenarios, $q_g = q$ for the 5 groups and $q_g = 1$ for all the other variables. The contribution of a group to the variance term of Φ is proportional to its size, as the importance reduction of the features inside the groups is not taken into account. This behavior is more pronounced in Figure 6.4b (scenario 2) than in Figure 6.4a (scenario 1), as the variables inside correlated groups are more relevant, causing the selection variance of the correlated groups to be even smaller. The POGR index essentially behaves like Φ in these two first scenarios.

The Pearson's correlation measure Φ_{pears} exhibits another behavior: it starts higher than the other measures and gradually decreases. When the total number of features d tends to ∞ , the correlation $\rho_{i,j}$ between feature

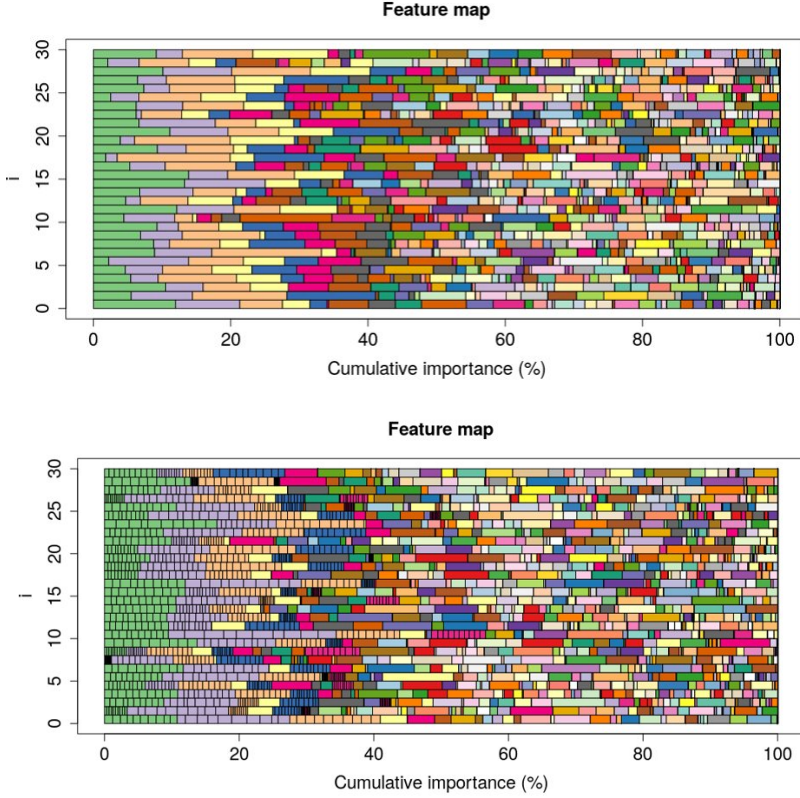


Figure 6.3: Feature stability maps of the group LASSO (scenario 1) when the size of the correlated groups c is equal to 1 (top) and 10 (bottom). The group LASSO regularization is chosen to select ≈ 40 features on average when $c = 1$, but the specific number of selected features here varies across runs. As the cumulative importance of each group is approximately constant in both feature stability maps, their stability should be similar.

importances of two runs i and j satisfies

$$\rho_{i,j} = \frac{\sum_f I_{i,f} I_{j,f}}{\sqrt{\sum_{f=1}^d I_{i,f}^2} * \sqrt{\sum_{f=1}^d I_{j,f}^2}}. \quad (6.8)$$

The contribution of a feature in the numerator of $\rho_{i,j}$ increases quadratically with its importance. When $q = 1$, it is dominated by the stability of the $N = 5$ feature groups, as their importance is the highest. Then, as q increases, the importance of each feature inside the groups is cut by q (as illustrated by Figure 6.3). Consequently, the sum of squared importance inside each group $\sum_{f \in g} I_{f,i}^2$ decreases by a factor q as well.

This implies an unintuitive result: the contribution of a correlated group to Φ_{pears} is inversely proportional to its size. When q grows, stability is more

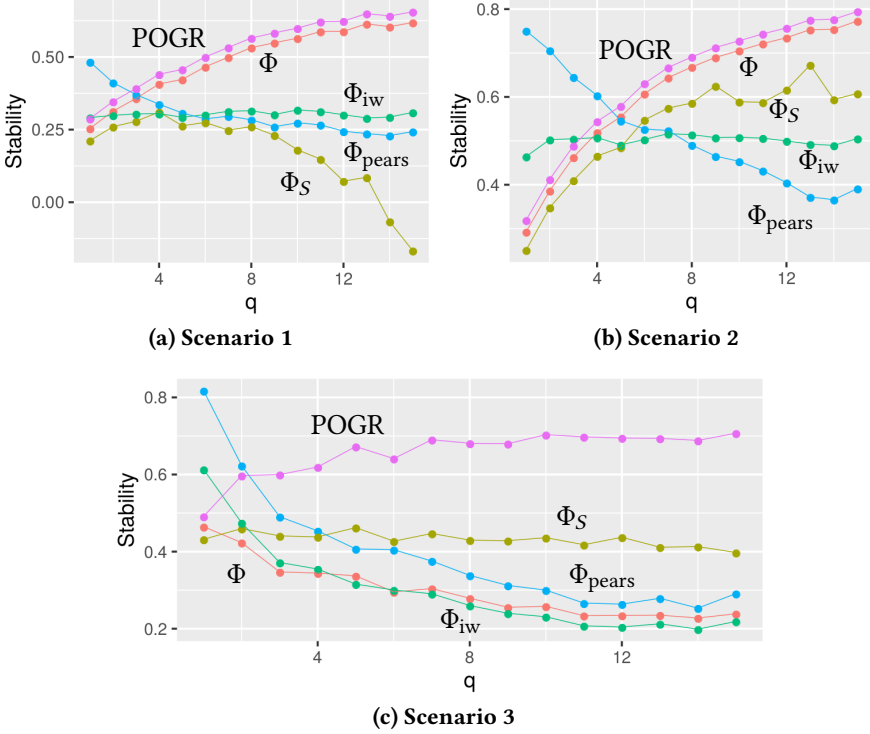


Figure 6.4: Experimental comparison of the stability measures Φ , Φ_S , Φ_{pears} , Φ_{iw} and POGR in the presence of highly correlated feature groups, in function of q , the size of such groups. The group LASSO is used for feature selection in (a)(scenario 1) and (b)(scenario 2), the LASSO in (c)(scenario 3). Given the design of these experiments, the stability value should not depend on q .

and more dominated by the out-of-group features, which have a larger selection variance. As was the case for Φ , this behavior is accentuated in Figure 6.4b (scenario 2), where the importance of correlated groups is larger (as is their relevance).

These results can be compared to the behavior of Φ_{iw} , where features contribute to the stability proportionally to their importance. Hence, as the sum of the importance of each group remains constant with respect to its size, so is their contribution to stability. As a consequence, Φ_{iw} remains approximately constant with c . Property 6 formalizes this result.

Property 6. *Indivisible group size independence: whenever perfectly correlated feature groups are selected as a whole, stability depends only on the groups' cumulative importance, not on the group sizes.*

As detailed in Appendix A.1, Property 6 strictly holds for Φ_{iw} if the global importance of each group is distributed among its features deterministically

across runs. Unlike the other measures, Φ_{iw} correctly assesses the relative importance of each group in the global stability value but may underestimate the within-group stability if the above assumption is violated.²

The evolution of Φ_S depends on the scenario considered. The experiments reported in Figure 6.4b are such that the correlated groups are sufficiently relevant for the classification so as to be selected in nearly all selection runs. In such a situation, Φ_S tends to increase with q . Whenever some correlated groups are regularly not selected, as in scenario 1 reported in Figure 6.4a, Φ_S actually tends to decrease with q and decays to $-\infty$.

Figure 6.4c is obtained with the pure LASSO selection which selects a few features inside correlated groups instead of the whole groups. In such a scenario, Φ , Φ_{pairs} and Φ_{iw} decrease with q as the selection of a few features inside each group becomes more and more unstable and because these measures do not take feature correlations into account. POGR somewhat increases with q while Φ_S remains approximately constant, which is the desired behavior.

6.4 Stability as Maximal Shared Importance

In this section, we introduce a novel stability measure and show that it behaves adequately in the two scenarios discussed in section 6.3.1. We further demonstrate that it generalizes previous work and prove several of its properties.

6.4.1 Definition

Unlike the limit cases presented in section 6.3.1, actual correlations between features need not be restricted to 0 or 1 values. Moreover, observing some correlation between a pair of features does not guarantee that they will necessarily share their importance values in models taking these features as input variables. Hence, to correctly assess stability while considering importance values, one needs to discover which features actually share importance values across predictive models built from different selection runs. This is why the evaluation of our stability index requires solving a linear program.

For each pair (i, j) of feature selection runs, one looks for the matching between features that maximizes a (similarity weighted) shared importance. Formally, $S(i, j)$ is the optimal objective value of the following constrained

²As shown in Figure 6.3, it is satisfied with the group LASSO. The global importance of each group is approximately evenly distributed among their features.

optimization problem:

$$S(i, j) = \max_{\mathbf{x}} \frac{\sum_{f,g} s_{f,g} \times x_{f,g}}{\bar{k}} \quad (6.9)$$

$$\text{subject to} \quad \sum_{g \in \mathcal{F}_j} x_{f,g} \leq I_{i,f}, \quad \forall f \in \mathcal{F}_i \quad (6.10)$$

$$\sum_{f \in \mathcal{F}_i} x_{f,g} \leq I_{j,g}, \quad \forall g \in \mathcal{F}_j \quad (6.11)$$

$$x_{f,g} \geq 0, \quad \forall (f, g) \in (\mathcal{F}_i, \mathcal{F}_j) \quad (6.12)$$

The variables $x_{f,g}$ represent the *latent* amount of shared importance by feature f , selected in run i , and feature g , selected in run j . This shared importance is multiplied by the similarity $s_{f,g}$ between the two features. The feature f can share its importance with several features of run j , but its total shared importance cannot exceed its own importance $I_{i,f}$ according to the constraint (6.10) (and its reciprocal (6.11)). Feature importance values are normalized such that $\sum_{f \in \mathcal{F}_i} I_{i,f} = \bar{k}$, the average number of selected features.³ Since the objective (6.9) and all constraints are linear with respect to the variables $x_{f,g}$, this optimization problem can be efficiently solved by linear programming. The stability over M feature selection runs is defined as the average pairwise optimal $S(i, j)$ values:

$$\Phi_{\text{msi}} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M S(i, j) \quad (6.13)$$

6.4.2 Illustration

An example of an optimal solution is depicted in Figure 6.5. Features 1, 2, 3 and 4 are selected in run i while features 2, 5, 6 and 7 are selected in run j . Feature 1 is heavily correlated to feature 6 ($s_{1,6} = 0.8$) and relatively well correlated to feature 5 ($s_{1,5} = 0.6$). Feature 6 is also somewhat correlated to feature 3 ($s_{3,6} = 0.4$). To maximize the objective, the link $x_{1,5}$ between features 1 and 5 is set to the maximum possible value, 0.7. The remaining importance of feature 1 is shared with feature 6 ($x_{1,6}^* = 0.6$), and the link between feature 3 and 6 is set to the maximum remaining importance of feature 6 ($x_{3,6}^* = 0.8$).

An intuitive analogy is to view the selected features and their importance as water containers, and the correlated links as water pipes. The objective is to assign the pipes to empty the containers above without overflowing the containers below, such that a maximum amount of water goes through the

³Since this normalization is undefined for a run i with $k_i = 0$ (a limit case with no feature selected in this run), we pose $S(i, j) = 0$ if $k_i = 0 \oplus k_j = 0$, and $S(i, j) = 1$ if $k_i = k_j = 0$, with $\oplus$ the XOR operator.

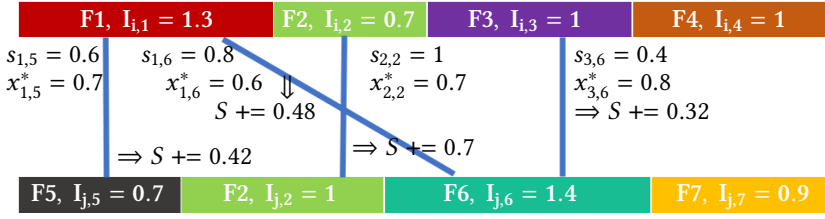


Figure 6.5: Example of optimal solution with $S(i, j) = \frac{0.42+0.48+0.7+0.32}{4} = 0.48$. One can verify that $x_{1,5}^* + x_{1,6}^* \leq I_{i,1} = 1.3$ and $x_{1,6}^* + x_{3,6}^* \leq I_{j,6} = 1.4$.

pipes with the highest weights (which are the similarities between features, or containers in the analogy).

The stability Φ_{msi} behaves correctly in the two scenarios presented in section 6.3.1. Considering the first scenario (Figure 6.1), the constant cumulative importance of the correlated group, independently of its size q , guarantees that the stability is also constant with q . Indeed, the optimal solution verifies $\sum_{q', q'' \in [1, q]} x_{q', q''} = \frac{\bar{k}}{4}$ which implies

$$S(i, j) = \frac{\sum_{q', q'' \in [1, q]} s_{q', q''} x_{q', q''} + \frac{\bar{k}}{4}}{\bar{k}} = \frac{\frac{\bar{k}}{4} + \frac{\bar{k}}{4}}{\bar{k}} = \frac{1}{2}, \forall q \geq 1,$$

as all the similarities $s_{q, q'}$ are equal to 1. In scenario 2 (Figure 6.2), the importance of feature 1 in the first run is shared among the q correlated features in the second run: $x_{1, q'} = I_{2, q'}$, for all $1 \leq q' \leq q$, the optimal objective value is then

$$S(i, j) = \frac{\sum_{1 \leq q' \leq q} s_{1, q'} x_{1, q'} + \frac{\bar{k}}{4}}{\bar{k}} = \frac{\sum_{1 \leq q' \leq q} I_{2, q'} + \frac{\bar{k}}{4}}{\bar{k}} = \frac{\frac{\bar{k}}{4} + \frac{\bar{k}}{4}}{\bar{k}} = \frac{1}{2}, \forall q \geq 1.$$

Section 6.4.4 further illustrates on simulated data that our measure Φ_{msi} does not suffer from the limitations of current measures.

6.4.3 Properties and Reductions

We show below that Φ_{msi} is bounded, which is necessary for a sound interpretation of the stability value, and fully defined (Property 7 and 8). We also demonstrate its maximality conditions in Property 9.

Property 7. *The stability measure Φ_{msi} is bounded in $[0, 1]$.*

Proof. As every variable $x_{f, g}$ and every entry of the similarity matrix S are positive, the objective (6.9) is positive as well. The measure is thus lower-bounded by 0.

If one assumes maximally similar features ($s_{f, g} = 1, \forall f, g$) instead of their actual similarity values, the resulting optimization problem has an optimal

objective value at least equal to the optimal objective value of the initial problem. The set of feasible solutions is the same as the constraints do not depend on $s_{f,g}$, and every solution has a larger or equal objective value than the corresponding solution of the initial problem. The optimal solution becomes

$$S(i, j) = \frac{\sum_{f \in \mathcal{F}_i} \sum_{g \in \mathcal{F}_j} x_{f,g}}{\bar{k}} \leq \frac{\sum_{f \in \mathcal{F}_i} I_{i,f}}{\bar{k}} = \frac{\bar{k}}{\bar{k}} = 1, \quad (6.14)$$

using constraint (6.10). The stability Φ_{msi} is thus upper-bounded by 1. $\square$

Property 8. *The stability measure Φ_{msi} is fully defined.*

Proof. We have posed $S(i, j) = 0$ if $k_i = 0 \oplus k_j = 0$ and 1 if $k_i = k_j = 0$, with $\oplus$ the XOR operator. It remains to show that the optimization problem always admits a feasible solution when k_i and k_j are both non-zero. Since $I_{i,f} \geq 0, \forall f \in \mathcal{F}_i, \forall i$, the trivial assignation $x_{f,g} = 0, \forall f, g$ is a solution. $\square$

Property 9. *The stability measure Φ_{msi} is maximal ($= 1$) iff, for all pairs of runs i and j , each feature importance in run i can be fully shared with the importance of one or several perfectly correlated features in run j . In other words, there exists no link $x_{f,g} > 0$ with $s_{f,g} < 1$, and all constraints from the set (6.10) are active: $\sum_{g \in \mathcal{F}_j} x_{f,g} = I_{i,f}, \forall f \in \mathcal{F}_i$.*

Proof. Suppose there exists a variable $x_{f,g} > 0$ with $s_{f,g} < 1$. Then increasing $s_{f,g}$ strictly increases the objective value. The previous objective value was thus not maximal. Suppose that $s_{f,g} = 1$ for every variable $x_{f,g} > 0$. Then, similarly to equation (6.14),

$$S(i, j) = \frac{\sum_{f \in \mathcal{F}_i} \sum_{g \in \mathcal{F}_j} x_{f,g}}{\bar{k}} \quad (6.15)$$

which is maximal ($= 1$) iff $\sum_{g \in \mathcal{F}_j} x_{f,g} = I_{i,f}, \forall f \in \mathcal{F}_i$. By reciprocity, the set of constraints (6.10) is active iff the set of constraints (6.11) is active. $\square$

Theorem 5 shows that solving the optimization problem (6.9) is equivalent to solving a maximum weighted bipartite matching problem whenever the importance of all selected features is evenly distributed between them in any given run and the number of selected features is constant across the M runs. In this specific case, Φ_{msi} reduces to the measure proposed by Yu et al. [YDL08].

Theorem 5. *Whenever the importance of all selected features is evenly distributed between them in any given run and the number of selected features is constant across the M runs, the constrained optimization problem (6.9) is a maximum weighted bipartite matching problem.*

Proof. The assumptions imply $I_{i,f} = 1, \forall f \in \mathcal{F}_i, \forall i$. Problem (6.9) has the form of $\{\max cx | Ax \leq b, x \geq 0\}$, with A the matrix of constraints (6.10) and (6.11). We first show that A is totally submodular, i.e., every of its square submatrix has determinant 0, +1 or -1. The following four conditions are sufficient for a matrix to be totally submodular [HT56]:

1. Every entry in A is 0, +1, or -1;
2. Every column of A contains at most two non-zero (i.e., +1 or -1) entries;
3. If two non-zero entries in a column of A have the same sign, then the row of one is in B and the other in C , with B and C two disjoint sets of rows of A ;
4. If two non-zero entries in a column of A have opposite signs, then the rows of both are in B or both in C , with B and C two disjoint sets of rows of A .

Condition 1 is satisfied as the coefficients multiplying $x_{f,g}$ in the set of constraints (6.10) and (6.11) are 0 or 1. Condition 2 holds because each variable $x_{f,g}$ has a non-zero coefficient in two constraints, one from the set of constraints (6.10) and one from (6.11). Condition 3 holds as we let B be the rows of A representing the set of constraints (6.10) and C be the rows of A corresponding to the set of constraints (6.11). If two non-zero entries in a column of A have the same sign, then one row represents a constraint in set (6.10) (and is thus in B) while the other represents a constraint in set (6.11) (and is thus in C). Condition 4 is trivially satisfied as two non-zero entries of A never have opposite signs.

If A is totally submodular and b is integral (which is here the case as $I_{i,f} = 1, \forall f \in \mathcal{F}_i, \forall i$), then linear programs of the forms $\{\max cx | Ax \leq b, x \geq 0\}$ have integral optimal solutions. In our case, the variables $x_{f,g}$ can only belong to $\{0, 1\}$ which makes the original optimization problem equivalent to maximum weighted bipartite matching. $\square$

If there is no feature correlation, i.e., if the similarity matrix is the identity matrix, theorem 6 shows that our measure Φ_{msi} simply computes the average pairwise feature importance intersection $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}$ between two selection runs, as defined in Section 5.3.1. This implies that whenever $\mathcal{S} = \mathbb{1}$ and the number d of input features tends to infinity, $\Phi_{\text{msi}} = \Phi_{\text{iw}}$.

Theorem 6. *Whenever the similarity matrix is the identity matrix,*

$$\Phi_{msi} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j|_{iw}}{\bar{k}}.$$

Proof. As $\mathcal{S} = \mathbb{1}$, the only variables $x_{f,g}$ that can contribute to the objective (6.9) are the variables $x_{f,f}$ linking the same feature in both selection runs. Variable $x_{f,f}$ exists only if feature f is selected in both runs. In this case, according to the constraints (6.10) and (6.11), $x_{f,f} \leq \min(I_{i,f}, I_{j,f})$, $\forall f \in \mathcal{F}_i \cap \mathcal{F}_j$. The equality is strict such as to maximize the objective. The objective can be restated as

$$S(i, j) = \frac{\sum_{f \in \mathcal{F}_i \cap \mathcal{F}_j} x_{f,f}}{\bar{k}} = \frac{\sum_{f \in \mathcal{F}_i \cap \mathcal{F}_j} \min(I_{i,f}, I_{j,f})}{\bar{k}} = \frac{|\mathcal{F}_i \cap \mathcal{F}_j|_{iw}}{\bar{k}}. \quad (6.16)$$

□

Theorem 7 states that Φ_{msi} turns into the normalized average feature intersection when the conditions of Theorem 5 and Theorem 6 are met.

Theorem 7. *Whenever the importance of all selected features is evenly distributed between them in any given run, the number of selected features is constant across the M runs and the similarity matrix is the identity matrix,*

$$\Phi_{msi} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j|}{\bar{k}}.$$

Proof. The proof comes back to proving that $|\mathcal{F}_i \cap \mathcal{F}_j|_{iw} = |\mathcal{F}_i \cap \mathcal{F}_j|$ in that scenario, which is done for Theorem 4 in Appendix A.1. □

Computing $S(i, j)$ requires to solve a linear programming problem with $k_i k_j$ variables, which can currently [Jia+20] be done in $O((k_i k_j)^{2.055})$ time. The overall time complexity to compute Φ_{msi} from feature importance⁴ is then

$$O(M^2 \overline{(k_i k_j)^{2.055}}) \approx O(M^2 \bar{k}^{4.11}).$$

Even though this is a somewhat high computational cost, it does not depend on the typically very high number of features d , unlike Φ_S that requires $O(d^2)$ time.

⁴As detailed in Section 5.3.2, computing feature importance can be done in $O(M\bar{k}n)$ in general and in $O(M\bar{k})$ for linear models.

6.4.4 Simulation Studies II

In this section, we revisit the simulations performed in Section 6.3.2, focusing on the behavior of Φ_{msi} .

Figure 6.6 shows that Φ_{msi} is the only stability measure to remain approximately constant in all the studied scenarios. The importance weighted measure Φ_{iw} behaves correctly in the two first scenarios, where the group LASSO is used as the feature selector, thanks to how it deals with feature importance. More specifically, this is a consequence of the measure verifying the *indivisible groups size independence* property (Property 6 stated in Section 6.3.2) under the condition that the global importance of each group is distributed among its features deterministically. The measure Φ_S can also remain constant in the last scenario, where the LASSO is used. The LASSO selects only a few features inside each group, and Φ_S can remain constant as it deals correctly with feature correlations in this instance. However, it does not depend on feature importance and thus fails to remain constant in the first two scenarios. The novel measure Φ_{msi} stays constant in all three scenarios because it deals both with feature importance and feature correlations. Much like Property 6 which targets scenarios 1 and 2, Property 10 formalizes this result. Any measure satisfying Property 10 should remain constant in the three discussed scenarios.

Property 10. *Within-group independence: whenever the feature space contains perfectly correlated feature groups, stability depends only on the groups' cumulative importance in each selection run, not on the number, identities, and relative importance of the selected features in each group.*

Property 10 is an extension of Property 6. As such, meeting Property 10 ensures meeting Property 6. The measure Φ_{msi} verifies Property 10 provided that $s_{f,g} = 1 \Rightarrow s_{f,f'} = s_{f',g} \forall f, g, f'$. This transitivity condition holds for the Pearson's and Spearman's correlation. Whenever it is respected, sharing the importance of features from the same group has an equal effect on the objective. As long as the total importance of the group is constant (hence the total importance that this group can share, according to the problem constraints), the objective value remains unchanged.

Tables from 6.3 to 6.8 summarize the properties satisfied by the different stability measures under study. The interchangeable property in these tables simply states that the selections of perfectly correlated features are interchangeable without any impact on stability.

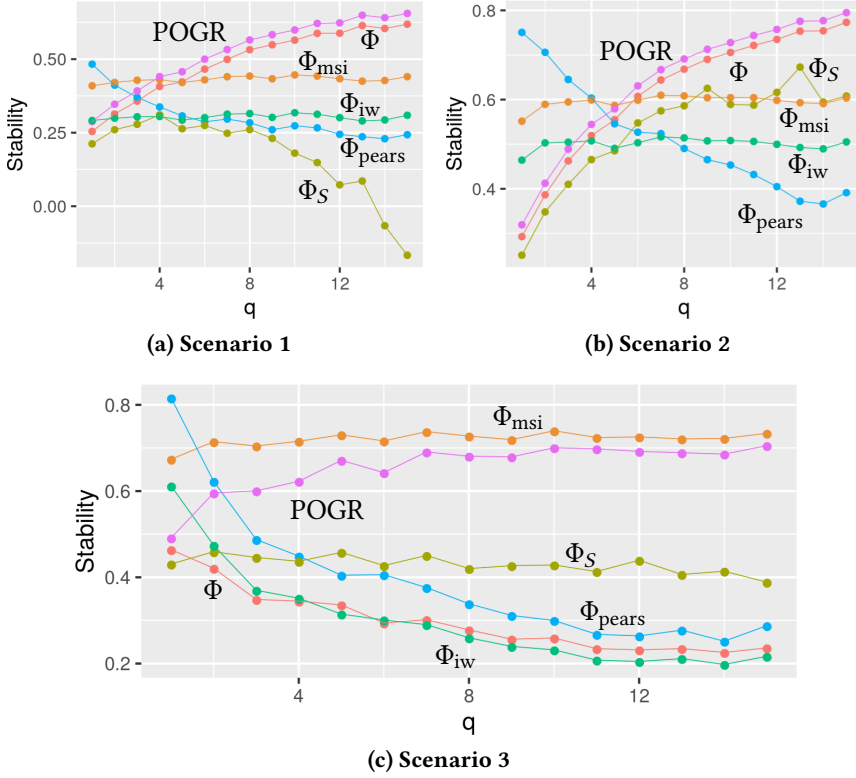


Figure 6.6: Experimental comparison of the stability measures Φ , Φ_S , Φ_{pears} , Φ_{iw} , POGR and Φ_{msi} in the presence of highly correlated feature groups, in function of q , the size of such groups. The group LASSO is used for feature selection in (a)(scenario 1) and (b)(scenario 2), the LASSO in (c)(scenario 3). Given the design of these experiments, the stability value should not depend on q .

Nogueira's Index Φ		
Property		
Fully defined	✓	
Upper bound	✓	≤ 1
Lower bound	✓	$\geq \frac{-1}{M-1}$
Maximum	✓	$\Leftrightarrow$ Deterministic selection
Corrected for H_0	✓	
Importance weighted decomposition	✗	Non -weighted decomposition
Indivisible group size independence	✗	The contribution of a group to Φ is proportional to its size
Interchangeable correlated features	✗	Function of the identities of the selected features only
Within-group independence	✗	Depends on the identities and number of selected features from each group

Table 6.3: Summary of the properties verified by the Nogueira's Index Φ . The measure cannot grasp nuances brought by feature importance and do not take variable correlation into account.

Pearson's correlation Φ_{pears}		
Property		
Fully defined	✓	
Upper bound	✓	≤ 1
Lower bound	✓	$\geq \frac{-1}{M-1}$
Maximum	✗	$\Leftrightarrow$ Linear dependence
Corrected for H_0	✓	
Importance weighted decomposition	✗	No general decomposition
Indivisible group size independence	✗	The contribution of a group to Φ_{pears} decreases with its size
Interchangeable correlated features	✗	Does not take feature similarity into account
Within-group independence	✗	Depends on the relative importance of the selected features from each group

Table 6.4: Summary of the properties verified by the Pearson's correlation measure Φ_{pears} . The measure takes feature importance into account in a way that undesirable behaviors can occur (notably in the presence of large correlated groups of variables). Furthermore, it does not satisfy the maximum property as only a perfect linear dependency is sufficient to make the measure maximal.

Importance Weighted Index Φ_{iw}		
Property		
Fully defined	✓	
Upper bound	✓	≤ 1
Lower bound	✓	$\geq \frac{-1}{M-1}$
Maximum	✓	$\Leftrightarrow$ Deterministic importance
Corrected for H_0	✓	
Importance weighted decomposition	✓	
Indivisible group size independence	$\pm$	The contribution of a group to Φ_{iw} is independent of its size (provided importance is distributed deterministically within each group)
Interchangeable correlated features	✗	Does not take feature similarity into account
Within group independence	✗	Depends on the relative importance of the selected features from each group

Table 6.5: Summary of the properties verified by the Importance Weighted Index Φ_{iw} . This index satisfies the *indivisible group size independence* and *importance weighted decomposition* properties which makes it more robust in different scenarios. Still, Φ_{iw} lacks the ability to consider highly correlated features as interchangeable (the alternate selection of highly correlated features creates instability).

POGR Index		
Property		
Fully defined	✓	
Upper bound	✓	≤ 1
Lower bound	✓	≥ 0
Maximum	✓	$\Leftrightarrow$ Systematic existence of a significantly correlated feature
Corrected for H_0	✗	
Importance weighted decomposition	✗	
Indivisible group size independence	✗	The contribution of a group to POGR is proportional to its size
Interchangeable correlated features	✓	Provided the significantly correlated relation is transitive
Within-group independence	✗	Depends on the number of selected features from each group

Table 6.6: Summary of the properties verified by the POGR Index. Note that the authors also propose a corrected for chance alternative, nPOGR, that loses the lower bound property [Zha+09]. This relationship between correction for chance and lower bound is discussed further in Section 6.3.

Sechidis Index Φ_S		
Property		
Fully defined	✓	
Upper bound	✗	
Lower bound	✗	
Maximum	✗	
Corrected for H_0	✓	
Importance weighted decomposition	✗	No general decomposition
Indivisible group size independence	✗	Complex
Interchangeable correlated features	✓	Provided $s_{f,g} = 1 \Rightarrow s_{f,f'} = s_{f',g} \forall f, g, f'$
Within-group independence	✗	Depends on the number of selected features from each group

Table 6.7: Summary of the properties verified by the Sechidis Index Φ_S . This measure extends Φ to deal with feature correlations at the expense of the bounds and maximum property.

Maximal Shared Importance Index Φ_{msi}		
Property		
Fully defined	✓	
Upper bound	✓	≤ 1
Lower bound	✓	≥ 0
Maximum	✓	$\Leftrightarrow$ Fully shared importance (Property 9)
Corrected for H_0	✗	
Importance weighted decomposition	$\pm$	Provided the selected features of the q methods are totally non-similar (between methods)
Indivisible group size independence	✓	The contribution of a group to Φ_{msi} is independent of its size
Interchangeable correlated features	✓	Provided $s_{f,g} = 1 \Rightarrow s_{f,f'} = s_{f',g} \forall f'$
Within-group independence	✓	Provided $s_{f,g} = 1 \Rightarrow s_{f,f'} = s_{f',g} \forall f'$

Table 6.8: Summary of the properties verified by the Maximal Shared Importance Index Φ_{msi} . This index combines feature importance and feature correlation and satisfies the *within group independence* property. It lacks the correction for chance property (see Appendix D for an alternative).

6.5 Stability of Standard Selection Methods

In this section, we report (accuracy, stability) Pareto fronts on real data sets, including microarray and mass spectrometry data. We compare the standard stability Φ , which ignores feature correlations, and Φ_{msi} . The similarity matrix $\mathcal{S}$ is estimated as the absolute values of the pairwise Spearman's ρ correlation: $s_{f,f'} = |\rho_{f,f'}|$. We consider the following feature selection methods and report performances over $M = 100$ selection runs.

- Random forests with 1000 trees. A first forest is learned on the original d features. The 20 features whose removal cause the greatest accuracy decrease on the out-of-bag examples are selected. A second forest of 1000 trees built on those 20 features is used for prediction.

- Logistic regression with a LASSO and ELASTIC-NET penalty. For the ELASTIC-NET $\lambda_1(\lambda_2 L_1 + (1 - \lambda_2)L_2)$, the parameter λ_2 , which dictates the balance between L1 and L2 norms, is set to 0.8. The parameter λ_1 is set to select, on average, $\bar{k} = 20$ features over the M runs.
- The RELIEF algorithm with 5 neighbors with equal weights to select 20 features. Predictive models are 5-NN classifiers.
- The logistic or hinge loss RFE. Predictive models are obtained by logistic regression or by fitting a linear SVM on the 20 selected features.
- The t-test, Mutual Information Maximization (MIM) (infotheo R package) and minimum Redundancy Maximum Relevance (mRMR) (mRMRe R package) information theoretic methods. For these three last approaches, the final model used for prediction is logistic regression estimated from the 20 selected features.

We report experiments on 5 microarray data sets (alon, singh, chiaretti, gravier and borovecki) from the datamicroarray R package and one mass-spectrometric data set, arcene, from the UCI machine learning repository. They have a high number of features (from 2,000 for alon to 22,283 for borovecki) with respect to the number of training examples (from 31 for borovecki to 198 for arcene), which generally leads to instability. In each selection run, the feature space is pre-filtered by keeping the 5,000 features with the highest variance before running a specific selection algorithm.

Figures 6.7 and 6.8 represent the (accuracy, stability) Pareto fronts across all feature selection methods for the two stability measures Φ (left) and Φ_{msi} (right). Results are also summarized in Table 6.9. These results show that the choice of best-performing feature selection methods highly depends on the stability measure used. Figure 6.7, Figure 6.8 and Table 6.9 further show that the stability Φ_{msi} is much higher than Φ for all selection methods. This phenomenon is the most pronounced on borovecki where $0.06 < \Phi < 0.24$ and $0.71 < \Phi_{\text{msi}} < 0.88$. This indicates that the observed instability is largely due to the high correlation between the input features. Even though the selection is unstable at the level of the input features, selected features from different selection runs tend to be highly correlated.

Correcting for the correlation between features, as done by Φ_S , produces unexpected results. Stability values according to Φ_S are detailed on Figure 6.9 and Table 6.10, with even lower values than those of Φ . This behavior contradicts the initial objective of addressing stability *underestimation* in correlated space. The measure Φ_S even often takes negative values that are hard to interpret. Section 6.6 studies this issue in more detail.

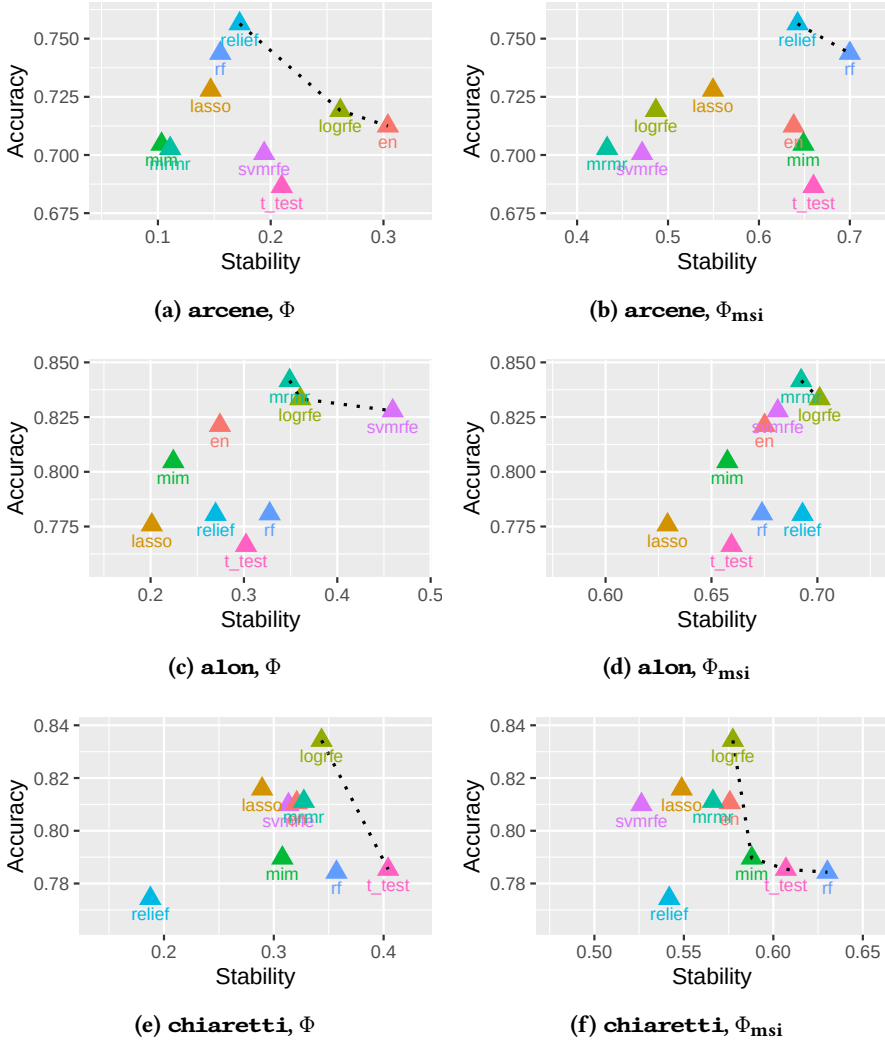


Figure 6.7: Pareto fronts across selection methods obtained when Φ (a,c,e) or Φ_{msi} (b,d,f) estimates stability, on **arcene** (a,b), **alon** (c,d) and **chiaretti** (e,f). These Pareto fronts depend on the choice of stability measure. On **arcene**, the random forests (rf), which were in the bottom half in terms of stability according to Φ , are the most stable method after correction for feature correlations. The logistic-rfe (logrfe) has lost the most. On **alon**, the svm-RFE loses its place on the Pareto-front, while on **chiaretti**, the mutual information maximization (mim) and the random forests (rf) are now on it. The relief has also caught up to the other methods.

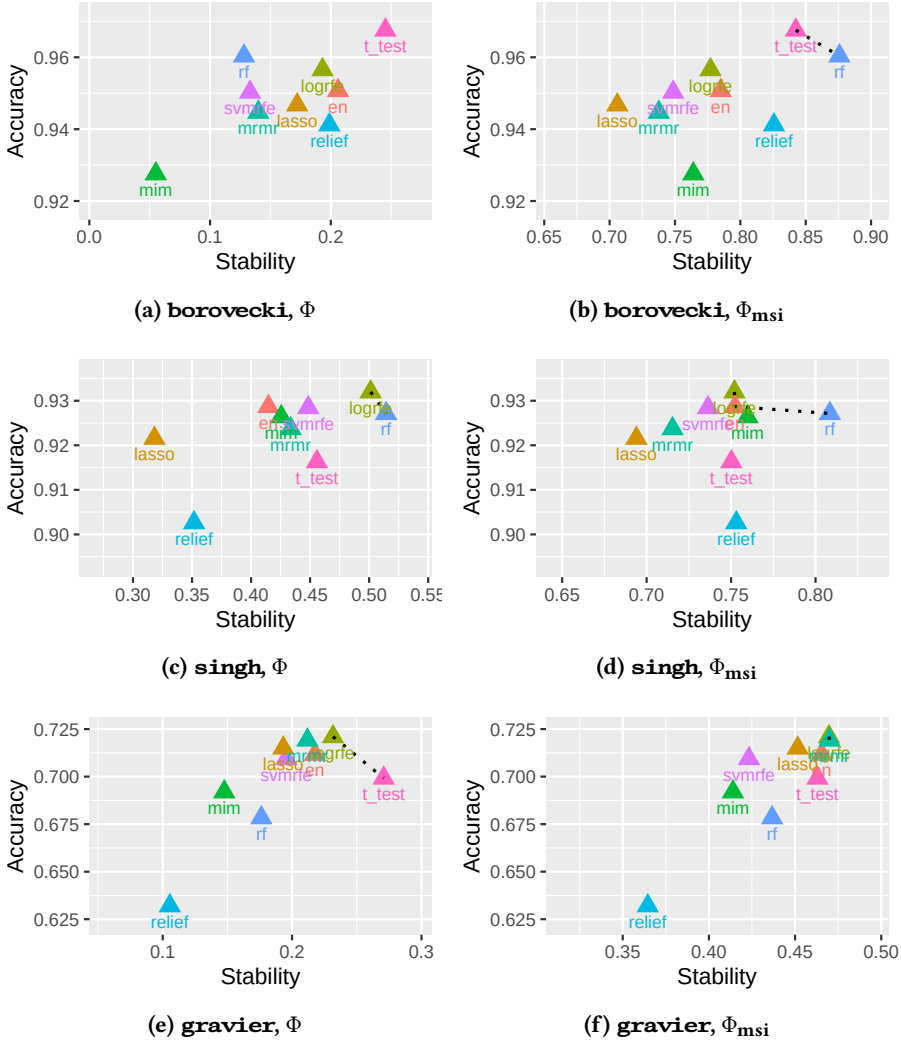


Figure 6.8: Pareto fronts across selection methods obtained when Φ (a,c,e) or Φ_{msi} (b,d,f) estimates stability, on **borovecki** (a,b), **singh** (c,d) and **gravier** (e,f). These Pareto fronts depend on the choice of stability measure. On **borovecki**, the random forests (rf) have the second-to-last worst stability according to Φ but is the most stable method considering Φ_{msi} . Mutual information maximization has also climbed up the order. On **singh**, the elastic-net joins the Pareto-front while the relief and the lasso reduce their deficit. On **gravier**, mRMR takes the place of the ttest on the Pareto-front.

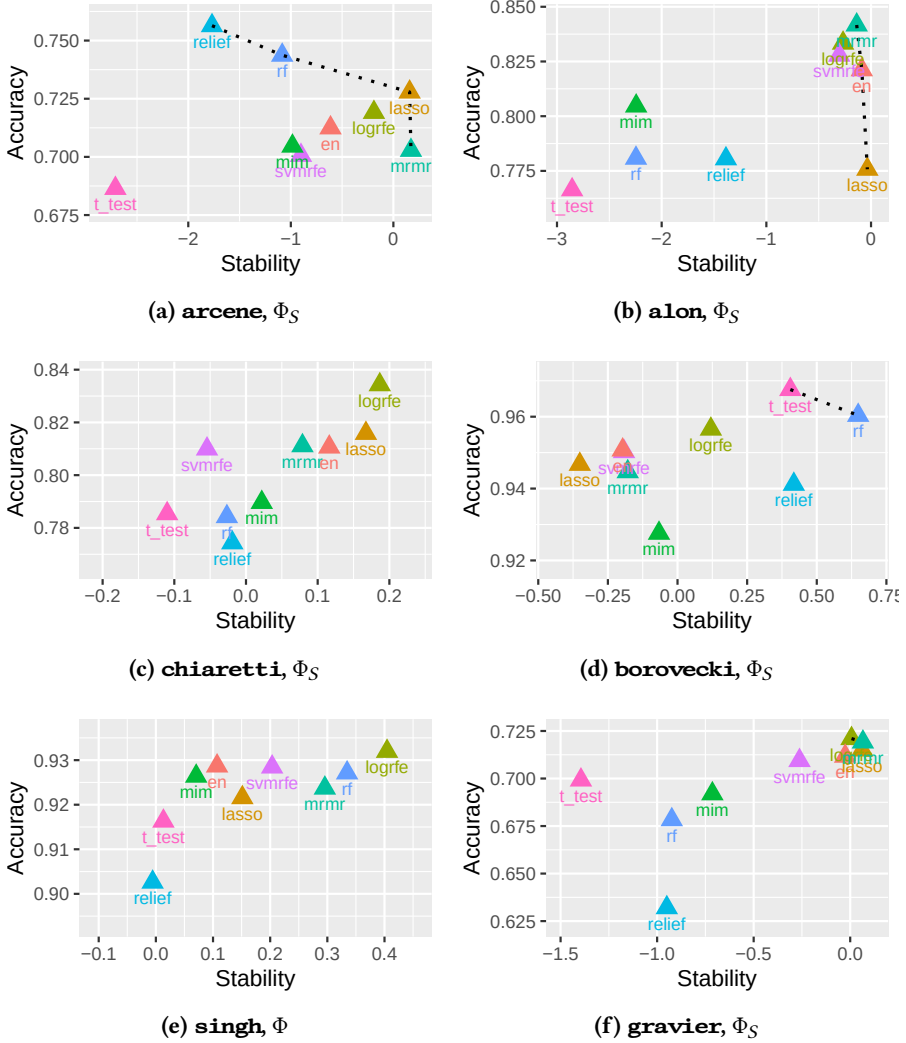


Figure 6.9: Pareto fronts across all selection methods, obtained when Φ_S estimates stability.

Data set	Range Φ	Pareto front Φ	Range Φ_{msi}	Pareto front Φ_{msi}	Range a_b
arcene	0.11-0.31	relief/logrfe/en	0.43-0.70	relief/rf/	0.69-0.76
alon	0.2-0.46	mrmr/logrfe/ svmrfe	0.63-0.70	mrmr/logrfe	0.77-0.84
chia.	0.19-0.4	logrfe/ttest	0.53-0.63	logrfe/mim/ ttest/rf	0.77-0.83
borov.	0.06-0.24	ttest	0.71-0.88	ttest/rf	0.93-0.97
singh	0.32-0.51	logrfe/rf	0.69-0.81	logrfe/en/rf	0.90-0.93
gravier	0.1-0.27	logrfe/ttest	0.36-0.47	logrfe/mrmr	0.63-0.72

Table 6.9: Stability ranges and Pareto fronts for Φ and Φ_{msi} on all data sets.

Data set	Range Φ_S	Pareto front Φ_S
arcene	[-2.71,0.43]	relief/rf/lasso/mrmr
alon	[-2.86,-0.04]	mrmr/en/lasso
chiaretti	[-0.11,0.19]	logrfe
borovecki	[-0.35,0.65]	ttest/rf
singh	[-0.01,0.4]	logrfe
gravier	[-1.4,0.07]	logrfe/mrmr

Table 6.10: Stability ranges and Pareto fronts for Φ_S on all data sets. Stability is often largely negative which creates interpretability issues.

6.6 The issue with the Sechidis Index

The Sechidis Index has been introduced in Section 6.2.1. By combining equation (6.4) and (6.5), this index can be expressed as

$$\Phi_S = 1 - \frac{\bar{k}_S - \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \mathbf{z}_i \mathbf{S} \mathbf{z}_j}{\text{tr}(\mathbf{S} \Sigma^0)}. \quad (6.17)$$

The measure increases with $\frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \mathbf{z}_i \mathbf{S} \mathbf{z}_j - \frac{1}{M} \sum_{i=1}^M \mathbf{z}_i \mathbf{S} \mathbf{z}_i$. The first term of this quantity estimates how similar are, on average, two selected feature sets from different runs i and j , according to the definition $|\mathcal{F}_i \cup \mathcal{F}_j|_S = \mathbf{z}_i \mathbf{S} \mathbf{z}_i$ (Equation 6.3). The second term computes the average of the similarity between identical runs ($\bar{k}_S$). Unlike for all the other studied measures, this average similarity between identical runs is not necessarily equal to $\bar{k}$ (and then normalized to 1). The quantity $\bar{k}_S$ is actually not upper-bounded, which is precisely what causes the measure to tend to $-\infty$ when highly correlated features are selected in the same selection run.

Consider the following example, depicted in Figure 6.10. Features 2, 3, and 4 are selected in both selection runs. Two perfectly (within-group similarity of 1) correlated groups of q features, in orange and red, are selected in the first and second run, respectively. We here assume that the total number of

features d tends to ∞ and that there exists no feature correlation outside of the correlated groups, like in Section 6.3.1, such that $\text{tr}(\mathcal{S}\Sigma^0) = \bar{k}$. In particular, we assume that $q < d$. In this scenario,

$$\bar{k}_S = \frac{1}{M} \sum_{i=1}^M \mathbf{z}_i \mathcal{S} \mathbf{z}_i = \frac{1}{2} \times 2 \times (q^2 + 3) = q^2 + 3.$$

The similarity of a selection run with itself increases with q^2 . We also have

$$\frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \mathbf{z}_i \mathcal{S} \mathbf{z}_j = 3.$$

The numerator of the fraction in equation (6.17) is equal to $q^2 + 3 - 3 = q^2$. Then,

$$\Phi_S = 1 - \frac{q^2}{\bar{k}} = 1 - \frac{q^2}{q+3},$$

which again tends to $-\infty$ when $q \rightarrow \infty$. Clearly, the issue here is that $\bar{k}_S$ has a squared dependency on the size q of the groups. This is caused by the scalar product, which counts the correlation between every pair of features. In contrast, our measure Φ_{msi} only depends on the correlation between matched features. In general, $\bar{k}_S$ increases a lot when highly correlated features are selected in the same selection run, which makes Φ_S likely to take unbounded negative values.



Figure 6.10: Scenario 3. Features 2, 3 and 4 are selected in both selection runs. Two correlated groups of q features, in orange and red, are selected in the first and second run, respectively.

We emphasize here that the primary issue is not related to feature importance. Ideally, in Figure 6.10, stability should be equal to $\frac{3}{4}$ for all q values as the cumulative importance of the correlated groups is constant, like in Figures 6.1 and 6.2. The expected value of a stability measure that ignores feature importance is here

$$\Phi = \frac{3}{q+3},$$

which is the exact value of Φ , KI, POGR or even Yu's index (introduced in Section 6.2.3). For the specific case where $q = 3$, all these measures are equal to $\frac{1}{2}$ (as one half of the features are selected in both runs), whereas Φ_S is already negative and equal to $-\frac{1}{2}$.

6.7 Conclusion

This chapter focuses on estimating stability in feature spaces with strong feature correlations. Feature selection algorithms tend to alternate between the selection of different, yet highly correlated features. Ignoring feature correlations leads to a pessimistic stability estimation.

Several correlation-aware stability measures exist in the literature, but they do not consider feature importance. Their limitations are illustrated theoretically, on toy scenarios, and on simulated data. The measure Φ_{iw} already improves the behavior of current measures whenever highly correlated feature groups are selected as a whole (e.g., with the group lasso) thanks to the way it deals with feature importance. Yet, it remains unsatisfactory in general when handling feature correlations.

A novel measure, Φ_{msi} , which takes into account both feature importance and feature correlation, is proposed. We define stability as the optimal objective value of a constrained optimization problem, which can be efficiently solved through linear programming. This objective depends on a similarity measure between features and their relative importance values in predictive models. We demonstrate on handcrafted examples and simulated data that our approach provides more relevant stability estimates than existing stability measures.

Experimental results on microarray and mass spectrometry data show that including feature correlations into the stability value generally changes the Pareto-optimal front. Consequently, that may strongly affect the choice of selection method when picking an optimal trade-off between feature selection stability and predictive performance.

The measure Φ_{msi} still lacks the property of correction for chance. Appendix D discusses this issue and evaluates different solutions. In particular, a corrected for chance variant of Φ_{msi} , Φ_{msi}^{cfc} is introduced. The variant comes with a higher computational cost and loses the lower-bound property. Experimental results suggest that the corrected for chance variant Φ_{msi}^{cfc} used in conjunction with a sparse similarity matrix might be the best choice. One could also define stability as the maximum value of Φ_{msi}^{cfc} over all sparsity thresholds, as defined in Appendix D. The resulting measure is strictly lower-bounded and asymptotically corrected for chance but suffers from even higher computational requirements.

Interactive Feature Stability Maps: an R package

7

This chapter introduces the *stabilitymaps* R package, which provides an interactive tool for domain analysis related to feature selection. The R package and its complete documentation is publicly available on this [link](#).

7.1 Overview

The goals of this package are twofold:

1. helping domain experts understand why an algorithm is found to be (un)stable and the implications
2. easing the identification of the most interesting features relative to a specific prediction task (such as the diagnosis of a disease based on an individual's gene expression profile)

This package quantifies the stability of machine learning algorithms that select a few features (e.g., gene expressions) and subsequently build predictive models on them for a specific task (e.g., the diagnosis of breast cancer). Some *perturbations* are applied to the data to form M slightly different versions of the original data set (different perturbation methods are available, we no longer restrict ourselves to bootstrapping). The same feature selection algorithm is run and produces a predictive model on each of these M modified data sets. Figure 2.17 in Chapter 2 depicts such a protocol. Ideally, the M selected feature sets should be highly similar. Unfortunately, this is often not the case, and we rely on a stability measure to quantify how robust to small perturbations are the feature preferences of the algorithm. In this context, the most interesting features are the ones with:

1. A high selection frequency. Features selected in most of the M selection runs are identified reliably by the algorithm, which seems confident about their relevance to the prediction task. The applied perturbations are insufficient to make the algorithm depart from these features.

2. A high importance in the predictive models. Predictive models do not always use their features in the same manner. Features that are the most important for prediction (the ultimate goal) are certainly the ones that matter the most in the involved (biological) processes.

The package further performs some feature similarity analysis. This helps identify features not selected with a high frequency but still representative of interesting underlying processes. For instance, if the selection alternates between two highly similar features (e.g., two genes with the same biological functions), both features might fail criteria 1) above. Still, the underlying biological functions are discovered often and might be of interest.

The package relies on the feature stability maps, a visualization tool first introduced in Chapter 5. These maps strongly facilitate the identification of the most promising features according to the two criteria listed above. Furthermore, they allow the intuitive estimation of stability according to our novel stability Φ_{iw} . The maps are here updated to show the solution of the constrained optimization problem that is solved to compute Φ_{msi} , easing the understanding of this second and more complex measure.

Reminder on how feature importance is computed In each selection run, a predictive model is built on the selected features to solve a particular predictive task. For linear models (e.g., logistic regression with a LASSO penalty), the importance of a feature is simply the absolute value of its weight in the model. Still, Section 5.3.2 proposes a non-linear generalization to evaluate the importance of a feature. In a nutshell, the smaller the change required to a feature for the model predictions to be affected, the more sensitive the model is to that feature, and the more important that feature is.

7.2 Demonstration on Leukemia diagnosis

The different functionalities of the package are demonstrated in this section on a real microarray data set (`chiaretti`). The predictive task is the diagnosis of leukemia.

STEP 1: load the data

To load the demonstration data set, simply type:

```
library("stabilitymaps")
data("sample_data")
```

This loads the leukemia data set into the `sample_data` R variable.

STEP 2: learn predictive models

This package proposes a baseline feature selection algorithm that can be used easily. Logistic regression with an elastic-net penalty is performed and produces linear models. To learn $M = 20$ of such linear models, type for instance:

```
r = elasticnets(sample_data$dataset,
                 sample_data$labels, M=20, lambda=0.1, alpha=1)
```

The `alpha` and `lambda` parameters are meta-parameters of the selection algorithm. Increasing `lambda` decreases the amount of selected features. The `alpha` parameter dictates the relative rate of L1 and L2 regularization (see Section 2.2.2.2 for a more detailed review of the elastic-net regression). The returned object `r` contains two attributes:

- `acc`, which is the average accuracy of the M predictive models
- `I`, a matrix with M (=number of selection runs) rows and d columns (=total number of features) containing the importance of each selected feature in the corresponding selection run. An importance of $I_{i,f} = 0$ indicates that the feature f is not selected in run number i .

The logistic-RFE can be used instead of the elastic-net by using the `logrfes` R function (see Section 2.2.2.4 for a review of the logistic-RFE algorithm).

What kind of perturbations are applied ? The default perturbation protocol (used throughout this thesis) is *bootstrapping*. Bootstrapping draws with replacement M samples of the same size as the original data set. A given data point has a probability of 63.2% to be present at least once in a given bootstrap sample. A predictive model is learned on each of these *bootstrapped* training sets and evaluated on the *out-of-bags* examples. An alternative strategy is to sample a fixed fraction of the original data without replacement. This can be done with the arguments `perturbationType` and `sampleSize`. For instance,

```
rSample08 = elasticnets(sample_data$dataset, sample_data$labels,
                          M=20, lambda=0.1, alpha=1, perturbationType="sample",
                          sampleSize=0.8)
```

samples 80% of the data points from the original data set 20 times to form 20 training sets. The elastic-net selection method is applied to each of these training sets. The learned predictive models are then evaluated on each run's remaining 20% of data points. By using the `sampleSize` argument, the user can easily control the amount of perturbation applied to measure instability. A large sample size results in less perturbation and vice-versa.

STEP 3: launch the interactive tool

The following code launches the interactive tool on the matrix `r$I` (Figure 7.1). The resulting plot is displayed in Figure 7.1.

```
o = precompute(r$I, sample_data$dataset,
               abr_names = sample_data$abr_names, acc=r$acc)
showMap(o, comp="fun")
```

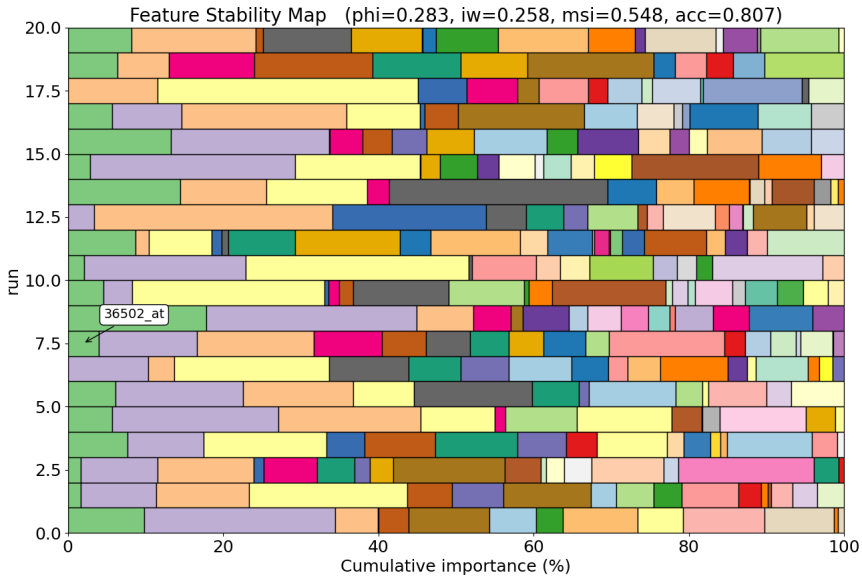


Figure 7.1: Feature stability map obtained with the demonstration data. Each row represents a given decision model, the whole map representing the $M = 20$ models. Each feature is assigned a color, and the rectangles' horizontal extensions measure the features' importance in the corresponding decision models. The features are sorted from left to right in decreasing order of selection frequency across runs. The green feature is selected the most often but has much less importance than the blue-grey, orange, and yellow ones, which might be more appealing for domain analysis.

Each row of these feature stability maps (introduced in Section 5.1) represents a given decision model, the whole map representing the M models learned over the M resamplings ($M = 20$ here). Each feature is assigned a color, and the rectangles' horizontal extensions measure the features' importance in the corresponding decision models, here linear ones estimated on each of such resamplings. For clarity, the features are sorted from left to right in decreasing order of selection frequency across runs.

As a domain analysis tool, these so-called feature stability maps allow the identification of the most reliably identified features (the most frequently se-

lected features, which are put at the left of such maps) and the most important features in the predictive models (the largest rectangles) that are subsequently built on the selected features. Features combining both properties are likely to be particularly appealing for domain experts. In Figure 7.1, the green feature (36502_at) is selected the most often (in 17/20 runs). Yet, it tends to have lower importance (two times less on average) in the predictive models than the other most selected features (the blue-grey, orange and yellow ones). Even though those three features are selected less often than the green one, they look more appealing for subsequent analysis as they actually matter more in the involved processes.

At the top of the stability map, three stability measures are displayed:

1. The Nogueira index Φ ($=\phi$)[NSB17a]. This measure is solely based on the selection frequencies of the selected features. It does not account for feature importance or feature similarity.
2. The importance weighted measure Φ_{iw} proposed in Section 5.3, which takes feature importance into account and generalizes Φ . Sections 5.6 and 6.3.2 demonstrate the advantages of using such a measure in the context of stability optimization and in the presence of groups of correlated features, respectively.
3. The measure Φ_{msi} , for maximal shared importance, proposed in Section 6.4. It further generalizes Φ_{iw} and also depends on feature similarities. Ignoring the possible similarities between features leads to a pessimistic estimation of the stability when some features are selected in one run and other features, distinct from but highly similar to the initial ones, are selected in another run. This measure solves this issue. The package allows domain experts to specify custom feature similarities and proposes automated solutions.

All these measures are asymptotically bounded in the interval $[0,1]$. In Figure 7.1, Φ is relatively low ($=0.283$): the selection algorithm is not confident about the identity of the features to use. The refined measure Φ_{iw} is even lower because the relative importance of the selected features also tends to change significantly from run to run. This is another source of instability that should be taken into account. Finally, Φ_{msi} , which accounts for feature similarities, is higher. Even though the selected features are vastly different from run to run, they tend to be somewhat similar to each other (according to a specific definition of similarity, explained later).

The name of the currently hovered feature is displayed. This name corresponds to the column names of the `I` argument given to the `precompute` function. These names (which have to be formatted if functional enrichment

is to be performed) can be overridden with the argument `abr_names`. One can search for a specific feature by entering its name in the search bar and clicking the bar.

When clicking on a feature in a given selection run m , the importance *shared* between this feature in this run m and all features in all the other runs is shown in grey. We encourage the reader to read again Section 6.1, which introduces the notion of *shared importance* intuitively, and Section 6.4, which formalizes it¹. The hatched part is the contribution this shared importance brings to stability, and is equal to the shared importance multiplied by the feature similarity². A demonstration feature map with only two runs is shown in Figure 7.2. In this example, the feature 38119_at has been clicked on. It matches the top-right feature on the bottom run. The total grey area in this top-right feature corresponds to the amount of *shared importance* between both features. In this case, it is equal to the total importance of 38119_at. The hatched part represents the contribution to stability caused by this matching. As both features are highly similar (0.8 here), both grey parts are nearly identical (i.e., selecting one or the other feature has only a small impact on stability). The algorithm seems unsure about the selection of both features. This makes them not very appealing for subsequent analysis at first sight. Nevertheless, the use of the proposed tool shows that the algorithm is confident that the underlying processes behind both features are pertinent. Naturally, this observation is not significant in this illustrative example as the number of runs M is set to 2.

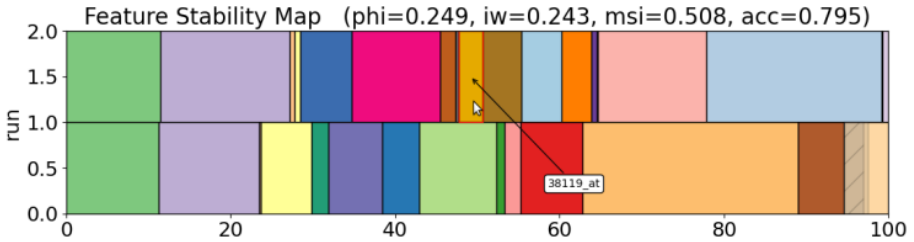


Figure 7.2: Small stability maps where the feature 38119_at has been clicked on. 38119_at matches the top-right feature on the bottom run. The total grey area in this top-right feature corresponds to the amount of shared importance between both features. The hatched part represents the contribution to stability caused by this matching. As both features are highly similar (0.8 here), both grey parts are nearly identical.

Coming back to the example of Figure 7.1, clicking on the black feature 1636_g_at in the run number 13 (starting at 0) gets us Figure 7.3. One can

¹The shared importance between two features f and g is the value of the variable $x_{f,g}$ of the solution of the optimization problem 6.9

²This contribution is $s_{f,g} \times x_{f,g}$ in Equation 6.9, up to the normalization constant $\bar{k}$.

see that 1636_g_at in run 13 tends to match its importance in other runs with

- itself, to which it is perfectly similar by definition
- the yellow feature, to which it is somewhat similar (≈ 0.5)
- the brown feature, to which it is also perfectly similar

The pie chart in Figure 7.4 is also produced and indicates the following statistics, which are independent of the clicked run:

- 1636_g_at has a selection frequency of 0.45 (selected in 9 of the $M = 20$ runs). It is the seventh feature with the highest selection frequency.
- The total importance of 1636_g_at amounts to 4.469% of the whole map (4.469% of the map is covered by the black of 1636_g_at). It is the fifth feature with the highest total importance.
- The outer pie indicates the amount of importance that 1636_g_at matches with other features. It matches a big chunk of its importance with the brown feature and with itself (respectively 23.2% and 21.6%). It also matches a decent amount with the yellow feature (13.8%).
- The inner pie indicates the contribution to stability that the shared importance cause. Stability is increased the most by the matching of 1636_g_at with itself and with the brown feature, 1635_at, as they are perfectly similar. Even though it also matches a large amount of importance with the yellow feature, stability is not increased as much as both features are only somewhat similar (≈ 0.5).

The selection algorithm is more confident about the relevance of the biological processes behind 1636_g_at and 1635_at than it appears at first sight (in 16/20 runs, at least one of 1636_g_at and 1635_at is selected, compared to 9/20 for 1636_g_at and 7/20 for 1635_at). Both features are never selected together, but one of them is selected in most selection runs. A closer look at these features might reveal exciting properties that could have been missed without a similarity analysis.

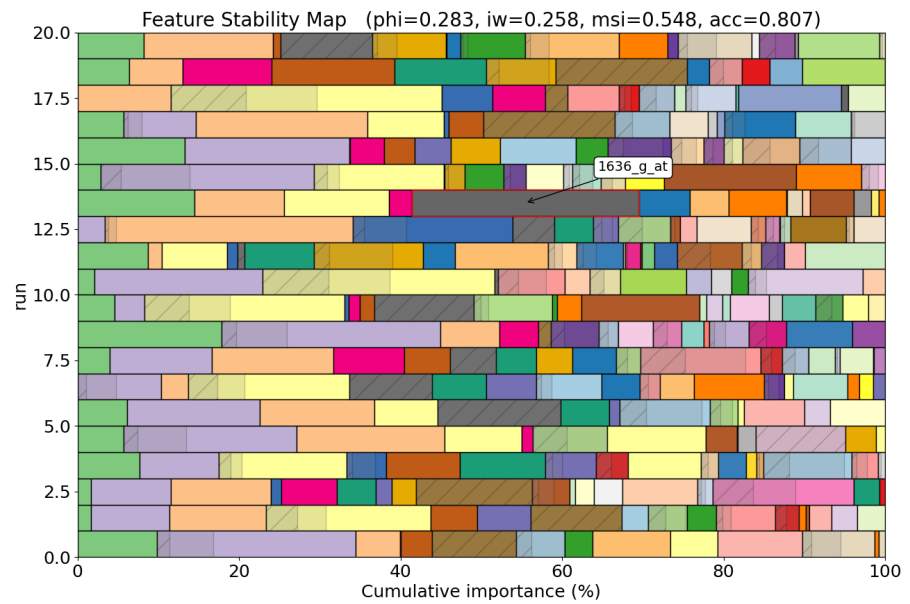


Figure 7.3: Feature stability map obtained with the demonstration data, when feature 1636_g_at in run number 13 has been clicked on. The grey areas correspond to the shared importance between 1636_g_at in run 13 and all other features in all other runs. The hatched parts correspond to the stability increases these matchings cause.

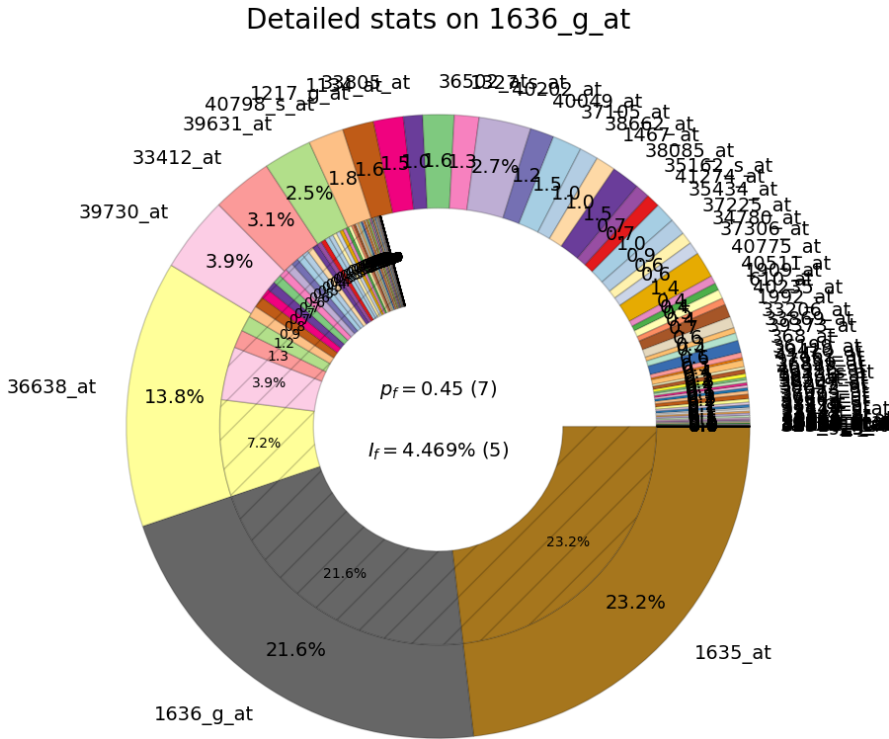


Figure 7.4: Pie chart obtained when feature 1636_g_at is clicked on. The outer rim indicates the amount of shared importance between 1636_g_at and all other features. The inner rim indicates the stability increase these matchings cause. Both rims are expressed in percentage of 1636_g_at's total importance. The selection frequency and total importance of 1636_g_at are displayed at the center of the pie, along with the ranks of 1636_g_at for both criteria.

What feature similarity to use ? The previously described similarity analysis naturally depends on the specific choice of similarity measure between features. The package proposes three of such similarity measures: 1) the absolute value of the Spearman's correlation, 2) the absolute value of the Pearson's correlation, and 3) a functional measure using the GeoSemSim R package [Yu+10]. This package derives a similarity measure between genes using the Wang graph-based measure [Wan+07] that estimates the similarity between two GO terms. This last measure has been used in this demonstration. The estimated stabilities are controlled by the `simiMethodsFeatures` argument of the function `precompute`. The package also allows sparsifying the similarity matrix for Spearman's and Pearson's similarity measures. This alleviates the issues caused by the absence of correction for chance of Φ_{msi} (see Appendix D). The corrected for chance variant $\Phi_{\text{msi}}^{\text{cfc}}$ can be evaluated instead of Φ_{msi} by setting the `chance` argument to `true`.

The user can specify his own similarity measure between features under the form of a matrix S such that $S[f][g]$ is the similarity between feature f and feature g . The following conditions on S must hold for stability to be well defined:

1. $0 \leq S[i][j] \leq 1$
2. $S[i][i] = 1$
3. $S[i][j] = S[j][i]$

The `simiMethodFeatures` argument must contain the "custom" string value and the `customSimilarityMatrix` argument must be set to S . For instance, the following code

```
o = precompute(r$I, sample_data$dataset,
  abr_names = sample_data$abr_names, acc=r$acc,
  simiMethodsFeatures = c("spearman-sparse", "fun", "custom"),
  customSimiMatrix = S)
```

computes stability according to three measures (the sparsified absolute value of Spearman's correlation, the functional measure, and the custom measure). To display the interactive stability maps, the choice of similarity measure must be specified by setting the `comp` argument of `showMap`:

```
showMap(o, comp="custom")
```

The package further allows studying the links between functional processes and selected features. To do so, the column names of the matrix I must be a valid keytype (such as ENTREZID, ENSEMBL, or SYMBOL), and this keytype must be specified by the `keytype` argument of `precompute`. The `funMappings` argument dictates which kind of functional analysis is performed:

- "enriched": only enriched GO terms are displayed
- "all": all GO terms annotated to the selected features are displayed

The following piece of code computes the results for both analyses and plots the one with all annotated terms (Figure 7.5):

```
o = precompute(r$I, sample_data$dataset,
  abr_names = sample_data$abr_names, acc=r$acc,
  funMappings=c("enriched", "all"))
showMap(o, comp="fun", funMapping="all")
```

One can see that the orange feature (38052_at) is annotated with several terms, some of which are clearly linked to blood. That certainly makes sense, as the prediction task is the diagnosis of leukemia.

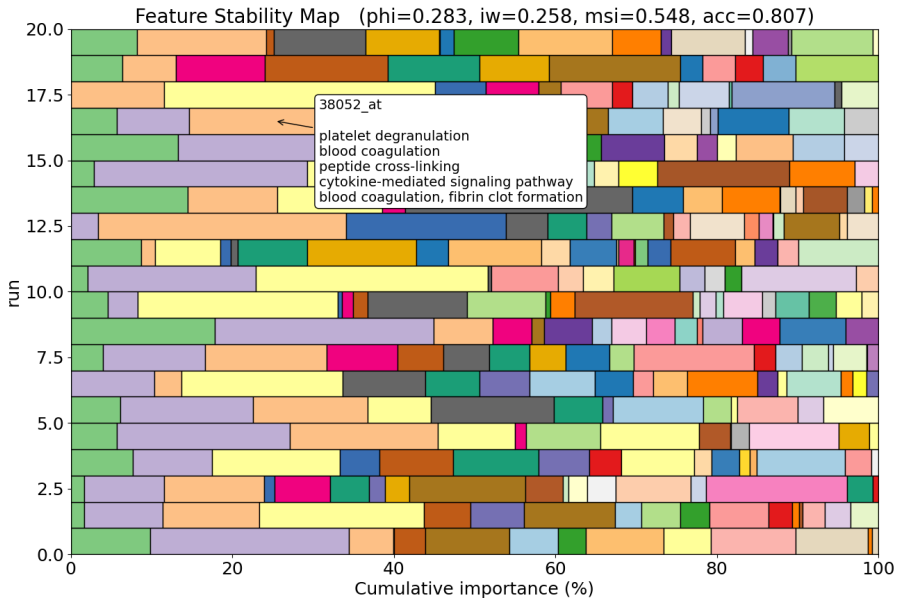


Figure 7.5: Feature selection stability map on demonstration data, with all GO terms annotated to the features displayed on hovering.

The following line launches the interactive tool in the GO terms space directly (Figure 7.6):

```
showMap(o, space="functions", funMapping="enriched")
```

By hovering over enriched GO terms, one can see to which features this term is annotated. Here, the GO term platelet degranulation is enriched in run 12 and is annotated to the features 35664_at, 36591_at, and the orange one 38052_at (all selected in run 12). The enriched GO terms are strongly unstable in this example. Note that the importance of a GO term g in run i is found as

$$I_{i,g} = \sum_{f:g \in GO(f)} \frac{I_{i,f}}{|GO(f)|},$$

with $GO(f)$ the set of (enriched) terms annotated to feature f . The similarity between GO terms is computed using the graph-based approach [Wan+07].

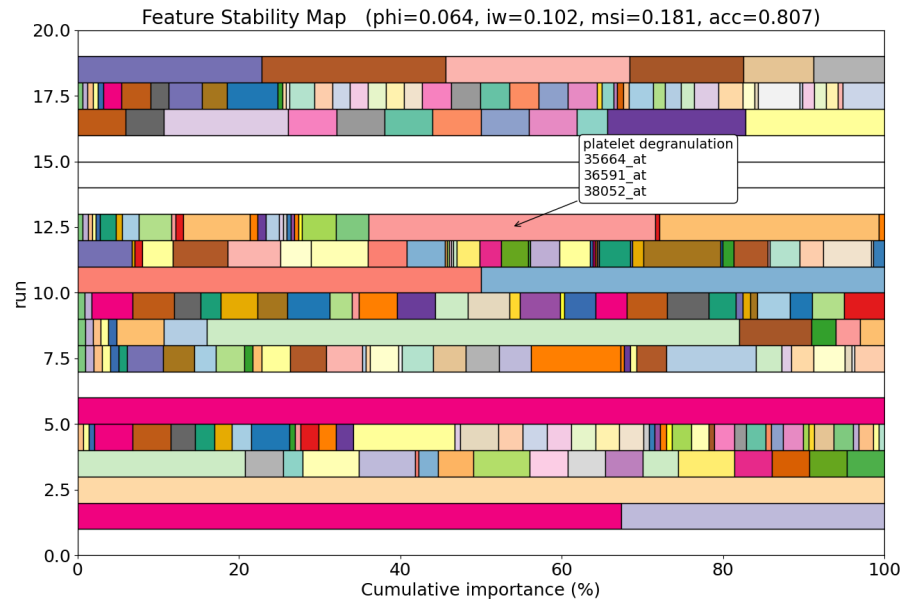


Figure 7.6: Feature selection stability map on demonstration data in the enriched GO terms space. The features (genes) to which is annotated the hovered GO term are displayed.

Conclusion and Perspectives

8

The typical instability of standard feature selection methods is a key concern nowadays as it reduces the interpretability of the predictive models and the trust of domain experts towards the selected feature subsets. Such experts would often prefer a more stable feature selection algorithm over an unstable but possibly slightly more accurate one. The lack of trust from domain experts certainly hinders the expansion of machine learning in sensitive areas such as the biomedical domain.

This thesis tackles two aspects of this stability issue: *stability optimization* and *stability estimation*.

Stability optimization is a field of research dedicated to improving selection stability. This thesis proposes the hybrid-RFE algorithm that optimizes stability explicitly via a dedicated meta-parameter. The algorithm first pre-selects some features based on a stable univariate criterion. In a second stage, it finds the most appropriate additional features in a multivariate fashion similar to the standard Recursive Feature Elimination algorithm. Results on microarray and mass-spectrometric data indicate that large stability gains are reached at a small cost of classification accuracy.

Perspectives on stability optimization The stability of the hybrid-RFE algorithm could potentially be improved further by combining it with some existing approach(es) targeting stability improvements. For instance, one could use ensemble feature selection [SAV08b; Abe+10] on top of the whole algorithm or on top of the univariate/multivariate selection separately. The data could also be pre-processed, for example, via instance-weighting for variance reduction [HY12].

Stability estimation seeks to measure the degree of stability of feature selection algorithms. Stability estimation should be made such that stability is an accurate indicator of the trust of domain experts towards the evaluated algorithms. Most of the contributions of this thesis fall into this research field.

The popular choice in the literature is to measure the stability of feature subsets independently of the feature's respective contributions to predictive

models. That option can have detrimental consequences. We illustrate this phenomenon in two situations: 1) when jointly optimizing selection stability with predictive performance and 2) when the feature space is composed of highly correlated feature groups. In these scenarios, selection algorithms may be regarded as untrustworthy by domain experts despite having high stability according to popular measures. This illustrates the importance of adequate stability estimation.

This thesis proposes two novel stability measures to solve the mismatch between stability and trustworthiness:

- The importance weighted measure Φ_{iw} incorporates the relative importance of the selected features in predictive models into the stability value. The proposed measure Φ_{iw} is fully-defined, bounded, corrected for chance, and maximal if and only if the importance distribution is deterministic. It is shown to improve decision-making when stability is jointly optimized with predictive accuracy. This is theoretically justified under the form of a new property, the *importance weighted decomposition* property, that Φ_{iw} satisfies. Using standard measures for stability estimation gives a false sense of stability. It may convince domain experts to sacrifice more predictive performance, even though the actual stability is not increased, as illustrated with the more robust measure Φ_{iw} . This is demonstrated experimentally on microarray and mass-spectrometric data.
- The maximal shared importance measure Φ_{msi} generalizes Φ_{iw} and deals with stability estimation in the presence of strong feature correlations. Stability is defined as the optimal objective value of a relaxed weighted bi-partite optimization problem, which can be efficiently solved by linear programming. The objective depends on a similarity matrix between features and their relative importance values in predictive models. The similarity matrix can be *a priori* defined or estimated from data. We demonstrate on handcrafted examples and simulated data that Φ_{msi} provides more relevant stability estimates than existing stability measures, including existing ones taking into account feature correlations. In its original formulation, Φ_{msi} lacks the correction for chance property. Appendix D proposes a corrected for chance variant of Φ_{msi} . The variant comes with an increased computational cost and at the expense of the lower-bound property. Experimental results suggest that sparsifying the similarity matrix improves the reliability of the corrected for chance variant.

We also propose a bi-objective evaluation framework to assess the aggregated quality of feature selection algorithms. The two objectives are prediction performance and selection stability. The framework is built on top

of a stability measure. It aims at measuring the average satisfaction of domain experts with respect to their favorite (accuracy, stability) trade-off from a specific Pareto-optimal front. Confidence intervals on the satisfaction of a domain expert are constructed. Such confidence intervals measure the uncertainty of the estimates due to the finite size of the data available and the particular data samplings used for assessment. Hypothesis tests to study the statistical significance of observed differences between two points or two Pareto fronts are also constructed. These tools should help data scientists compare different feature selection algorithms and favor the ones with higher average satisfaction.

Perspectives on stability estimation The evaluation framework, including the statistical tools, can be used with any satisfaction function. This can be seen as a strength or a weakness, as the specific values reported in this thesis are conditional to the particular choice of satisfaction function. Furthermore, assumptions on the variability of the subjective preference of domain experts for accuracy and stability are made. As detailed in Section 3.6, surveys on data scientists or actual domain experts could be made to relax or refine those assumptions.

The proposed evaluation framework has been introduced in the first methodological chapter of this thesis. It uses a pre-existing stability measure (Φ) to quantify stability. We recommend using our novel measure Φ_{iw} instead. Chapter 5 builds confidence intervals and hypothesis tests on the importance weighted measure's asymptotic value. This ensures that Φ_{iw} can be used instead of Φ in the evaluation framework. Doing so certainly improves the validity of the reported satisfaction estimates. The statistical tools have been constructed under the assumption that the measure's sampling distribution asymptotically tends to a normal distribution. Past work proves this result for Φ , and future work certainly includes the derivation of a similar proof for its generalization Φ_{iw} . Nonetheless, the normality assumption and the related statistical tools have been successfully validated on real data in a wide range of scenarios. This gives experimental evidence for asymptotic normality. We also let the derivation of confidence intervals on Φ_{msi} as future work. This might be challenging as the measure is formulated as the objective of the solution of a constrained optimization problem. Yet, asymptotic normality could also be assessed experimentally.

Computing Φ_{msi} requires solving a constraint optimization problem for each pair of runs. As detailed in Section 6.4.3, its computational complexity is $O((k_i k_j)^{2.055}) \approx O(M^2 \bar{k}^{4.11})$. This can become computationally intensive, especially when the number of selected features increases. In this thesis, we use the well-known simplex algorithm as a solver. One could study possible

ways to increase the speed of the solver, for instance, by developing a fast heuristic for the initial solution used by the simplex (or other more efficient solvers).

A domain analysis tool Chapters 3 to 6 study stability estimation and optimization. They are mostly directed to data scientists. Chapter 7 is domain expert oriented. It provides tools to the expert to facilitate domain analysis related to feature selection in the form of an R package. Feature stability maps (first introduced in Chapter 5) allow the intuitive estimation of stability according to Φ_{iw} . Chapter 7 adds an interactive layer to these maps. It displays, on demand and in an intuitive way, the solution of the constrained optimization problem that is solved to compute Φ_{msi} . These stability maps help domain experts understand why an algorithm is found to be (un)stable according to our novel stability measures. They also help domain experts identify the most interesting features relative to a specific prediction task. They allow the identification of the most frequently selected features and the features that contribute the most to the predictive models that are subsequently built. Features combining both properties are likely to be particularly appealing for domain experts. Through detailed statistics on feature correlations and matched importance, the package also helps identify features not selected with a high frequency but representative of interesting underlying processes. These biological processes and their selection stability can also be studied easily within the R package.

Perspectives on domain analysis In Chapter 6 and Appendix D, the feature similarity matrix on which Φ_{msi} is built has been estimated on data. The similarity between two features is defined as the (possibly thresholded) absolute value of their Spearman's correlation. The proposed R package notably allows using an *a priori* defined similarity measure. This functional measure uses the GeoSemSim R package [Yu+10] that derives a similarity measure between genes based on their GO annotations. The stability Φ_{msi} associated to this biological similarity matrix can be seen as a *functional* stability. Such a functional stability is measured at the level of the underlying biological processes rather than from the selected genes. Further research is needed in that direction. For instance, one could study the differences between the stability values and Pareto-fronts obtained with classical selection stability and with this refined functional stability (similar to the survey in Section 6.5). The impact of the correction for chance on this functional stability should be assessed as well (as done in Appendix D for the Spearman's correlation similarity matrix).

Further perspectives This thesis focuses on single-task stability. Perturbations are applied (here through bootstrapping) to the data of a single predictive task. Stability is also of interest in the *transfer learning* and *multi-task* settings. Multi-task selection algorithms perform feature selection on multiple tasks jointly. A feature set (and a predictive model) is learned on each specific predictive task. Stability should also be promoted in the multi-task setting as features relevant for multiple tasks should be particularly interesting to study. Standard multi-task feature selection cannot be performed when all learning tasks are not directly available. Nonetheless, information from the tasks arising first can be propagated to subsequent tasks via *transfer learning*. As for multi-task feature selection, stability has to be encouraged from the domain expert's point of view.

No information is shared between the M selection runs in the single-task stability setting. Stability with respect to small data variations is assessed. In the transfer learning and multi-task settings, stability is assessed over the M_t tasks. Promoting feature sharing over the different tasks is usually done through dedicated regularization schemes (e.g., Group-Lasso with ℓ_1/ℓ_p regularization [VR12], additive linear models [EP04], see [ZY17] for a complete survey). Such methods were introduced with the primary objective of building accurate predictive models across several tasks. A possible direction for future research is to study to which extent these methods could also be used to allow the tuning of the across-task selection stability and classification performance trade-off. Furthermore, it should be possible to explicitly include the objective of stability in the loss minimization problem that is solved in the multi-task and transfer learning settings. This could promote stability more directly than existing approaches. The measures Φ and Φ_{iw} are naturally non-convex with respect to the model parameters (e.g., the weight vector $\mathbf{w}$ in a linear model). Consequently, some proxy for these measures might be needed. Including Φ_{msi} in a loss function should be even harder as it is formulated as the objective of a constrained optimization problem.

Multi-task learning in the biomedical domain is motivated by recent advances in RNAseq technologies, for which gene expression can now be measured at the level of single cells. Finding relevant biomarkers, in this case, requires finding informative patterns in a collection of patients, each of them being represented by transcript data over hundreds of cells. A future research goal could be to study the stability of gene selection through multi-task learning for which the single-cell expression patterns of an individual patient would represent a specific task.

Additional proofs



In this Appendix, mathematical results developed in Chapter 5 are formally proven.

A.1 Proofs of Properties

This section proves the 6 properties satisfied by the importance weighted stability measure

$$\Phi_{\text{iw}} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} - |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}}{\bar{k} - C} \quad (\text{A.1})$$

with

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = \sum_{f \in \mathcal{F}_i \cap \mathcal{F}_j} \min(I_{i,f}, I_{j,f}), \quad |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = \frac{1}{d} \sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}),$$

and

$$C = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}.$$

These properties are summarized in Table 6.5.

Property 1. *Fully defined: the measure is defined for every possible importance combinations.*

Proof. For Φ_{iw} to be undefined, the corrective term C must be equal to $\bar{k}$, such that the denominator of Equation (A.1) is equal to 0. As feature importances are normalized such that $\sum_f I_{i,f} = \bar{k}, \forall 1 \leq i \leq M$, $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}$ is upper bounded by $\bar{k}$, $\forall i, j$. Indeed, such a normalization implies

$$\sum_{f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}) \leq \bar{k}, \forall f \in \mathcal{F}_i.$$

As there are at most d features selected in run i ,

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = \frac{1}{d} \sum_{f \in \mathcal{F}_i} \sum_{f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}) \leq \frac{1}{d} d \bar{k} \leq \bar{k}.$$

For the inequality not to be strict, all d features must be selected in both selection runs and the importances must all be equal such that $\min(I_{i,f}, I_{j,f'}) = I_{i,f} = I_{j,f'} \forall f, f'$ or no feature must be selected in both runs (as we posed $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}} = \bar{k}$ in this situations). It then follows that Φ_{iw} is well defined unless all features are always selected with a constant ($= 1$) importance in each run, or no feature is ever selected.

□

Property 2. *Maximum stability $\Leftrightarrow$ deterministic importance: the measure reaches its maximum if and only if the importance of every feature is constant across all runs.*

Proof. For Φ_{iw} to be maximal ($= 1$, see Property 3), $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}$ must be equal to $\bar{k} \forall i, j$. This condition is equivalent to $I_{i,f} = I_{j,f} \forall f, i, j$. In other words, Φ_{iw} is maximal when the importance of each feature is constant across selection runs. We refer to such a situation as deterministic. □

Property 3. *Bounds: the measure is bounded by constants not dependent on the overall number of features d or on the average number of features selected $\bar{k}$.*

Proof. Our measure Φ_{iw} , defined in Equation (5.1), can be trivially restated as

$$\Phi_{\text{iw}} = \frac{\left(\frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} \right) - C}{\bar{k} - C}.$$

As $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} \leq \bar{k} \forall i, j$, this implies $\frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} \leq \bar{k}$. The measure Φ_{iw} is thus upper bounded by 1. The proof of the lower bound is more complex. Referring to

$$A = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}},$$

we first use the chain of equivalence:

$$\begin{aligned} \Phi_{\text{iw}} \geq \frac{-1}{M-1} &\Leftrightarrow \frac{A-C}{\bar{k}-C} \geq \frac{-1}{M-1} \\ \Leftrightarrow \frac{M}{M-1}C - A &\leq \frac{\bar{k}}{M-1} \Leftrightarrow M^2C - M(M-1)A \leq M\bar{k}. \end{aligned}$$

We maximize $M^2C - M(M-1)A$ using linear programming and show that the optimal solution cannot exceed $M\bar{k}$.

$$\begin{aligned}
& \text{maximize} \quad \frac{M}{M-1} \frac{1}{d} \sum_i \sum_{j \neq i} \sum_f \sum_{f'} z_{ijff'} - \sum_i \sum_{j \neq i} \sum_f z_{ijff} \\
& \text{subject to} \quad z_{ijff'} - w_{if} \leq 0 \\
& \quad \quad \quad z_{ijff'} - w_{jf'} \leq 0 \\
& \quad \quad \quad z_{ijff} - w_{if} \geq 0 \\
& \quad \quad \quad z_{ijff} - w_{jf} \geq 0 \\
& \quad \quad \quad \sum_f w_{if} = \bar{k} \quad \forall i \\
& \quad \quad \quad z_{ijff'} \geq 0, \quad w_{if} \geq 0.
\end{aligned}$$

The first four constraints impose that $z_{ijff'} = \min(w_{if}, w_{jf'})$. As the sign of $z_{ijff'}$ in the objective is positive, at least one of the first two constraints will always be tight. The same is true for the 3rd and 4th constraints, as the sign of z_{ijff} is negative in the objective ($\frac{M}{M-1} \frac{1}{d} - 1 \leq 0$ with $d \geq 2, M \geq 2$). This leads to the following dual formulation.

$$\begin{aligned}
& \text{minimize} \quad \sum_i \bar{k} W_i \\
& \text{subject to} \quad y_{ijff'1} + y_{ijff'2} \geq \frac{M}{M-1} \frac{1}{d} \text{ for } f \neq f' \\
& \quad \quad \quad y_{ijff1} + y_{ijff2} \geq \frac{M}{M-1} \frac{1}{d} - 1 \\
& \quad \quad \quad - \sum_{j \neq i} \sum_{f'} y_{ijff'1} - \sum_{i \neq j} \sum_{f'} y_{ijff'2} + W_i \geq 0 \quad \forall i, f \\
& \quad \quad \quad y_{ijff'} \geq 0, y_{ijff} \leq 0.
\end{aligned}$$

The assignment $y_{ijff'} = \frac{M}{M-1} \frac{1}{2d}$, $y_{ijff} = \frac{M}{M-1} \frac{1}{2d} - \frac{1}{2}$, $W_i = M$ is a solution to the problem. The first two constraints are trivially satisfied. One can check that the third one is also satisfied:

$$- \sum_{j \neq i} \sum_{f'} y_{ijff'1} - \sum_{i \neq j} \sum_{f'} y_{ijff'2} + W_i = (-(M-1)d \frac{M}{M-1} \frac{1}{2d}) \times 2 + M = 0.$$

This solution has an objective value of $M\bar{k}$. Consequently, the optimal value of the primal formulation is upper bounded by $M\bar{k}$. $\square$

Property 4. *Correction for chance: the measure is constant in expectation (here set to 0) when features are selected randomly.*

Proof. The average contribution to $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}$ that a feature selected in both runs i and j imply is

$$\frac{\sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'})}{k_i k_j}.$$

If features were to be selected at random (*i.e.*, under the null model of feature selection H_0), there would be, on average, $\frac{k_i k_j}{d}$ of such jointly selected features. Then,

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{H_0} = \frac{1}{d} \sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}) = |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}.$$

Also, in the special cases where k_i , k_j , or both are equal to 0, $|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}$. Under H_0 , the numerator of Equation (5.1) is then equal to 0, making $\Phi_{\text{iw}}|H_0 = 0$. $\square$

In Section 5.4, we study the behavior of Φ , Φ_{pears} and Φ_{iw} when different feature selection methods are combined. Combining different selection methods allows the exploration of part of the (accuracy, stability) objective space whenever they produce different (accuracy, stability) trade-offs. This is illustrated in Section 5.6 with the hybrid-RFE (introduced in Section 4.1) which combines the classical logistic RFE and a more stable, yet less accurate univariate criterion. Property 5 states that the contribution of each of the combined selection methods to the stability should be proportional to the importance of their selected features in the global predictive model. We prove Property 5 for Φ_{iw} under the assumption that d tends to ∞ . One of the consequences of this result is that Φ_{iw} , unlike Φ , cannot be increased by the selection of stable, yet marginally important features (see Section 5.6 for practical evaluations).

Property 5. *Importance weighted decomposition: when combining the non overlapping selected feature sets of Q different methods, which produce Q models, in a single predictive model, if features selected by method q have their importance multiplied by δI_q in the combined predictive model of each selection run, stability can be expressed as a weighted sum of the Q prior stabilities:*

$$\Phi_{\text{iw}} = \sum_{q=1}^Q \frac{\bar{k}_q}{\bar{k}} \delta I_q \Phi_{\text{iw}}^q$$

with Φ_{iw}^q the stability of method q alone, and δI_q the factor such that $I_{i,f} = \delta I_q \times I_{i,f}^q$.

Proof. Assuming that d tends to ∞ , we can neglect the correction for chance terms, such that

$$\Phi_{\text{iw}} = \frac{2}{M(M-1)} \frac{\sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}}{\bar{k}}.$$

As there is no feature intersections between the sets selected by the Q methods, one can write

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = \sum_q \sum_{f_q} \min(I_{f,i}^q \delta I_q, I_{f,j}^q \delta I_q) \Rightarrow$$

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} = \sum_q \delta I_q \sum_{f_q} \min(I_{f,i}^q, I_{f,j}^q) = \sum_q \delta I_q |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^q.$$

Then,

$$\begin{aligned} \Phi_{\text{iw}} &= \frac{2}{M(M-1)} \sum_q \delta I_q \frac{\sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^q}{\bar{k}} \Rightarrow \\ \Phi_{\text{iw}} &= \sum_q \frac{\bar{k}_q}{\bar{k}} \delta I_q \frac{2}{M(M-1)} \frac{\sum_{i=1}^M \sum_{j=i+1}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^q}{\bar{k}_q} = \sum_q \frac{\bar{k}_q}{\bar{k}} \delta I_q \Phi_{\text{iw}}^q. \end{aligned}$$

To keep the normalization $\sum_f I_{i,f} = \bar{k}, \forall i$ correct, the δI_q values must verify

$$\sum_q \frac{\bar{k}_q}{\bar{k}} \delta I_q = 1$$

such that

$$\sum_f I_{i,f} = \sum_q \delta I_q \sum_{f_q} I_{f,i}^q = \sum_q \delta I_q k_q = \bar{k}.$$

□

Property 6 concerns the behavior of stability when the feature space is composed of highly correlated feature groups, which is studied in detail in Section 6.3.2. It formalizes why our proposed measure Φ_{iw} can remain approximately independent from the correlated groups sizes in the two first scenarios studied in that section (Figures 6.4a and 6.4b), where the group LASSO is used for feature selection. We prove that Φ_{iw} satisfies Property 6 under the following assumptions: 1) the importance of each group is deterministically distributed among their respective selected features, and 2) d tends to ∞ .

Property 6. *Indivisible group size independence: whenever perfectly correlated feature groups are selected as a whole, stability depends only on the groups' cumulative importance, not on the group sizes.*

Proof. According to the assumptions (1) and (2) above, it follows that the importance of feature f , which belong to group g , in selection run i , satisfies

$$I_{i,f} = I_{g,i}^1 \frac{\bar{k}}{\bar{k}^1 c_g} \delta_f \quad (\text{A.2})$$

with $I_{g,i}^1$ the importance of the group g in run i whenever the group size c_g is reduced to 1, $\bar{k}^1$ the average number of selected features in this reference scenario, and δ_f the constant fraction of importance feature f has in group g ($0 \leq \delta_f, \sum_{f \in g} \delta_f = 1$). One can easily verify that Equation (A.2) satisfy the normalization constraint

$$\sum_{f=1}^d I_{f,i}^1 = \bar{k}^1, \forall i \Rightarrow \sum_{f=1}^d I_{i,f} = \bar{k}, \forall i.$$

Let I_f^* be, as in Equation (5.8), equal to $\sum_{i,j=1}^M \frac{\min(I_{i,f}, I_{j,f})}{|z_{f,\cdot} \neq 0|^2}$, where $|z_{f,\cdot} \neq 0|$ is the number of runs where feature f is selected. Similarly, let $I_g^{*,1} = \sum_{i,j=1}^M \frac{\min(I_{g,i}^1, I_{g,j}^1)}{|z_{f,\cdot} \neq 0|^2}$. It follows that

$$\begin{aligned} I_f^* &= \frac{1}{|z_{f,\cdot} \neq 0|^2} \sum_{i,j} \min(I_{g,i}^1 \frac{\bar{k}}{\bar{k}^1 c_g} \delta_f, I_{g,j}^1 \frac{\bar{k}}{\bar{k}^1 c_g} \delta_f) \Rightarrow \\ I_f^* &= \frac{\frac{\bar{k}}{\bar{k}^1 c_g} \delta_f}{|z_{f,\cdot} \neq 0|^2} \sum_{i,j} \min(I_{g,i}^1, I_{g,j}^1) = \frac{\bar{k}}{\bar{k}^1 c_g} \delta_f I_g^{*,1}. \end{aligned}$$

Then, using Equation (5.8),

$$\sum_g \sum_{f \in g} I_f^* p_g^2 = \sum_g p_g^2 \frac{\bar{k}}{\bar{k}^1 c_g} \sum_{f \in g} \delta_f I_g^{*,1} = \sum_g p_g^2 \frac{\bar{k}}{\bar{k}^1} I_g^{*,1} \Rightarrow$$

$$\Phi_{iw} = 1 - \frac{\bar{k} - \sum_g \sum_{f \in g} I_f^* p_g^2}{\bar{k} - C} = 1 - \frac{\bar{k} - \frac{\bar{k}}{\bar{k}^1} \sum_g I_g^{*,1} p_g^2}{\bar{k}^{c_g} - C} = 1 - \frac{\bar{k}^1 - \sum_g I_g^{*,1} p_g^2}{\bar{k}^1 - C} = \Phi_{iw}^1,$$

with Φ_{iw}^1 the stability of the reference scenario where the group sizes c_g are all equal to 1. $\square$

A.2 Proofs of Theorems

This section proves Theorem 3, Theorem 4 and Equation (5.8).

Theorem 3. *Whenever the importance of all selected features is evenly distributed between them in any given run,*

$$\Phi_{iw} = \frac{\mu_M\left[\frac{\bar{k}|\mathcal{F}_i \cap \mathcal{F}_j|}{\max(k_i, k_j)}\right] - \frac{\bar{k}}{d} \mu_M\left[\frac{k_i k_j}{\max(k_i, k_j)}\right]}{\bar{k} - \frac{\bar{k}}{d} \mu_M\left[\frac{k_i k_j}{\max(k_i, k_j)}\right]}$$

with $\mu_M(g(i, j)) = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M g^*(i, j)$, $g^*(i, j) = 1$ if $k_i = k_j = 0$, $g(i, j)$ otherwise.

Proof. As feature importance is constant inside each selection run, according to the normalization $\sum_f I_{i,f} = \bar{k}, \forall i$, the importance of a feature in selection run i is equal to $\frac{\bar{k}}{k_i}$. Starting from $|\mathcal{F}_i \cap \mathcal{F}_j|_{iw} = \sum_f \min(I_{i,f}, I_{j,f})$, we get

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{iw} = \sum_{f \in \mathcal{F}_i \cap \mathcal{F}_j} \min\left(\frac{\bar{k}}{k_i}, \frac{\bar{k}}{k_j}\right) = \frac{\bar{k}|\mathcal{F}_i \cap \mathcal{F}_j|}{\max(k_i, k_j)}.$$

Also,

$$|\mathcal{F}_i \cap \mathcal{F}_j|_{iw}^{\text{rand}} = \frac{1}{d} \sum_{f \in \mathcal{F}_i, f' \in \mathcal{F}_j} \min(I_{i,f}, I_{j,f'}) = \frac{1}{d} k_i k_j \min\left(\frac{\bar{k}}{k_i}, \frac{\bar{k}}{k_j}\right) = \frac{\bar{k}}{d} \frac{k_i k_j}{\max(k_i, k_j)}.$$

$\square$

Theorem 4. *Whenever the importance of all selected features is evenly distributed between them in any given run and the number of selected features is constant across the M runs,*

$$\Phi_{iw} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j| - \frac{\bar{k}^2}{d}}{\bar{k} - \frac{\bar{k}^2}{d}},$$

which is the usual expression of the Kuncheva index.

Proof. Using Theorem 3 and posing $k_i = \bar{k} \forall i$, we get $\frac{\bar{k}}{\max(k_i, k_j)} = 1$ and $\frac{k_i k_j}{d} = \frac{\bar{k}^2}{d}$. Then,

$$\Phi_{\text{iw}} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j| - \frac{\bar{k}^2}{d}}{\bar{k} - \frac{\bar{k}^2}{d}},$$

which is the usual expression of the Kuncheva index. As the Kuncheva index is equivalent to Φ when the number of selected features is constant, so is Φ_{iw} . $\square$

We further prove the frequency-based restatement of Φ_{iw} (Equation 5.8):

$$\Phi_{\text{iw}} = 1 - \frac{\frac{M}{M-1}(\bar{k} - \sum_{f=1}^d I_f^* p_f^2)}{\bar{k} - C} = 1 - \frac{\sum_{f=1}^d I_f^* s_f^2 + \frac{M}{M-1}(\bar{k} - \sum_f I_f^* p_f)}{\bar{k} - C}$$

with p_f the selection frequency of feature f in the M runs, $s_f^2 \triangleq \frac{M}{M-1} p_f(1-p_f)$ its selection variance and $I_f^* = \sum_{i,j=1}^M \frac{\min(I_{i,f}, I_{j,f})}{|z_{:,f} \neq 0|^2}$ a properly normalized global feature importance, where $|z_{:,f} \neq 0|$ is the number of runs where feature f is selected. In the following proof, $z_{i,f}$ is the Bernoulli variable indicating the selection or non-selection of feature f in selection run i .

Proof. We first develop the average pairwise similarity between selection runs:

$$\begin{aligned} \frac{1}{M(M-1)} \sum_{i=1, j \neq i}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} &= \frac{1}{M(M-1)} \sum_{i=1, j \neq i}^M \sum_{f=1}^d \min(I_{i,f}, I_{j,f}) z_{i,f} z_{j,f} \\ &= \frac{1}{M(M-1)} \sum_{i,j=1}^M \sum_{f=1}^d \min(I_{i,f}, I_{j,f}) z_{i,f} z_{j,f} - \frac{1}{M(M-1)} \sum_{i=1}^M \sum_{f=1}^d I_{i,f} z_{i,f} z_{i,f} = \\ &= \frac{1}{M(M-1)} \sum_{f=1}^d M^2 I_f^* p_f^2 - \frac{\bar{k}}{M-1} = \frac{M}{M-1} \sum_{f=1}^d I_f^* p_f^2 - \frac{\bar{k}}{M-1} = \\ &= \frac{M}{M-1} \sum_{f=1}^d I_f^* p_f - \sum_{f=1}^d I_f^* s_f^2 - \frac{\bar{k}}{M-1}. \end{aligned}$$

Injecting into Φ_{iw} ,

$$\Phi_{\text{iw}} = \frac{2}{M(M-1)} \sum_{i=1, j > i}^M \frac{|\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}} - |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}^{\text{rand}}}{\bar{k} - C}$$

$$\begin{aligned}
&= \frac{\frac{1}{M(M-1)} (\sum_{i=1, j \neq i}^M |\mathcal{F}_i \cap \mathcal{F}_j|_{\text{iw}}) - C}{\bar{k} - C} \\
&= \frac{(\frac{M}{M-1} \sum_{f=1}^d I_f^* p_f - \sum_{f=1}^d I_f^* s_f^2 - \frac{\bar{k}}{M-1}) - C}{\bar{k} - C} \\
&= \frac{(\frac{M}{M-1} \sum_{f=1}^d I_f^* p_f - \sum_{f=1}^d I_f^* s_f^2 - \frac{M\bar{k}}{M-1}) + \bar{k} - C}{\bar{k} - C} \\
&= 1 - \frac{\sum_{f=1}^d I_f^* s_f^2 + \frac{M}{M-1} (\bar{k} - \sum_f I_f^* p_f)}{\bar{k} - C}.
\end{aligned}$$

□

Decision Making for Cancer Diagnosis

B

This Appendix studies further the behavior of the measures Φ , Φ_{pears} and our measure Φ_{iw} in the context of joint optimization with predictive accuracy. As detailed in Section 5.6, Φ is best increased by the selection of features that are stable, but possibly irrelevant and only marginally used for prediction. Optimizing Φ_{pears} can give highly varying importance to the selected features (example in Figures 5.11 and 5.12). Optimizing our proposed measure leads to more satisfactory feature stability maps with higher predictive performance.

This Appendix presents the results obtained with the hybrid-RFE on the data sets *borovecki* (Figures B.1 and B.2), *gravier* (Figure B.3), *chiaretti* (Figure B.4), *alon* (Figure B.5) and *arcene* (Figure B.6). As in Figures 5.11 and 5.12 in Section 5.6 (on *singh*), the plain points in the figures below represent all Pareto-optimal trade-offs when Φ (red), Φ_{pears} (green) and Φ_{iw} (blue) are used to assess stability. The circled points are the compromises $(\tilde{a}_b, \Phi)$ (red), $(\tilde{a}_b, \Phi_{\text{pears}})$ (green) and $(\tilde{a}_b, \Phi_{\text{iw}})$ (blue) that are obtained with meta-parameters on the $(\tilde{a}_b, \Phi)$ (a), $(\tilde{a}_b, \Phi_{\text{pears}})$ (b) and $(\tilde{a}_b, \Phi_{\text{iw}})$ (c) Pareto-optimal curves. The Pareto-optimal fronts obtained when each measure is optimized are in general different. This illustrates that optimizing Φ or (generally to a lesser extent) Φ_{pears} is not guaranteed to increase Φ_{iw} , which can lead to unsatisfactory results (see feature stability maps of Figures 5.11 and 5.12 for *singh* and Figure B.2 for *borovecki*). Even when Pareto-fronts are obtained with similar sets of meta-parameters, the γ values corresponding to each compromise can vary significantly (ex: *gravier* (Figure B.3)), which ultimately leads to different chosen trade-offs.

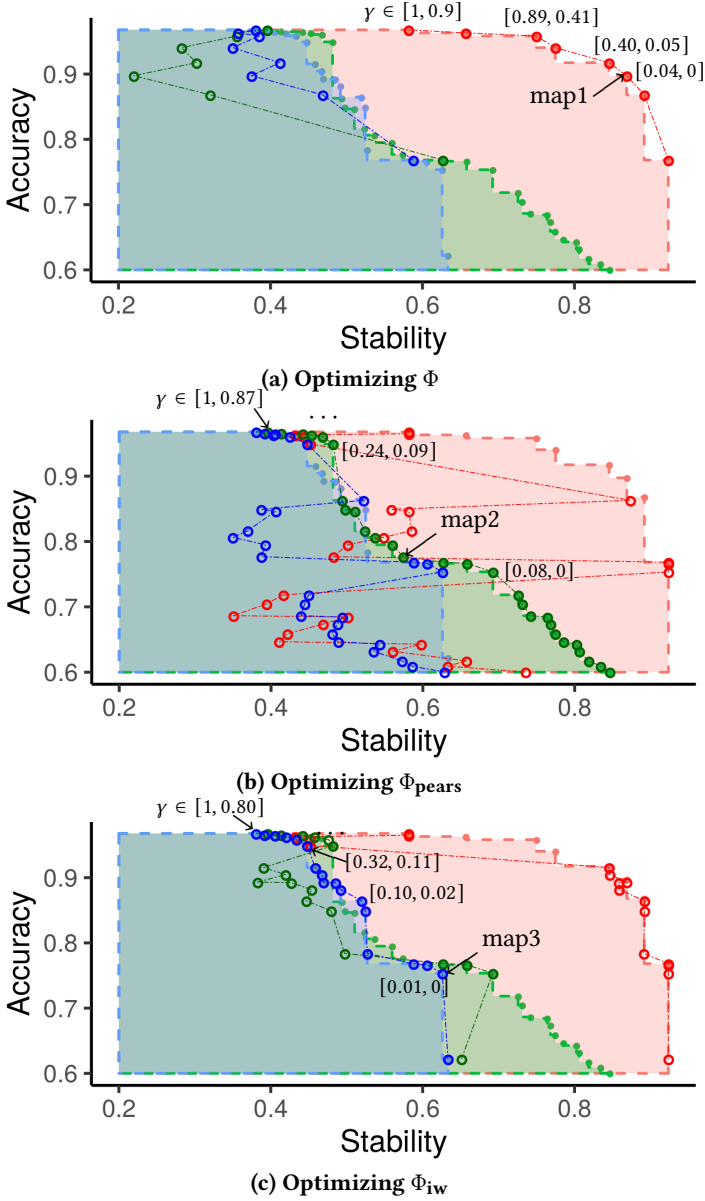


Figure B.1: Pareto-optimal curves for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on `borovecki`. From (a), optimizing Φ does not necessarily increase Φ_{pears} and Φ_{iw} . Furthermore, accuracy is sacrificed for γ values much bigger when Φ is optimized. Using Φ to estimate stability gives a false sense of stability and may convince domain experts to sacrifice more predictive performance, even though the more robust measure Φ_{iw} is not increased. Domain experts with a preference parameter $\gamma \in [0.04, 0]$ pick the point labeled **map1**. About 7% of accuracy is sacrificed even though Φ_{iw} is not improved. From (b), increasing Φ_{pears} is not guaranteed to increase Φ_{iw} either. The annotated feature maps are depicted in Figure B.2.

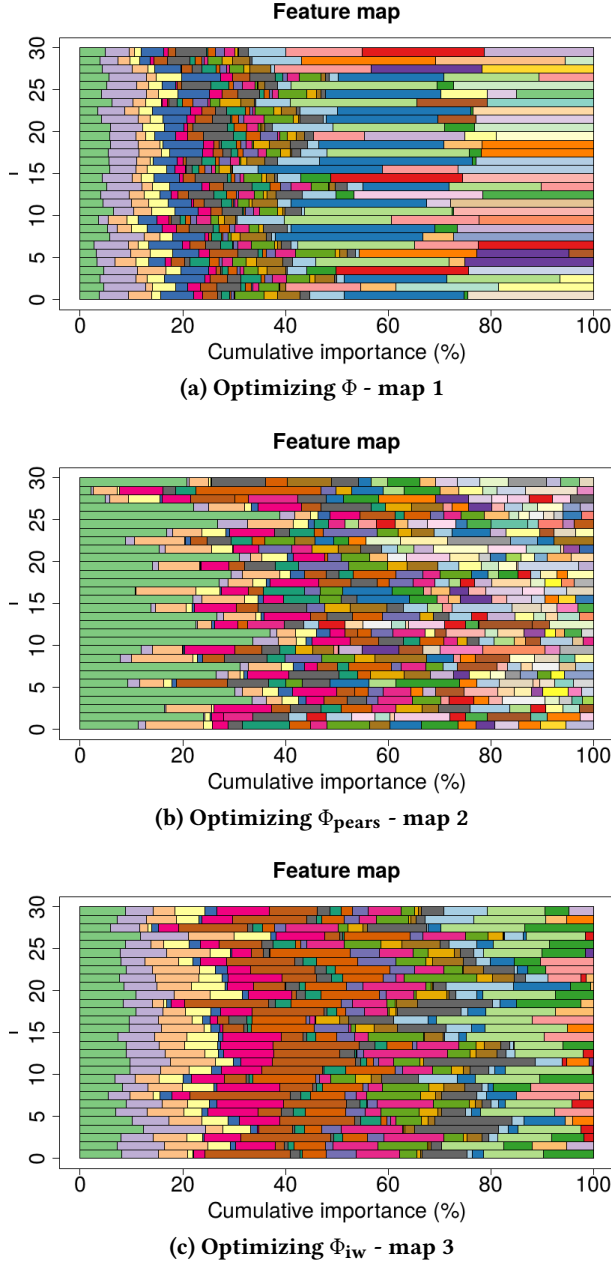


Figure B.2: Stability maps extracted from Figure B.1. Optimizing Φ results in the importance of most frequently selected features to be very small. The whole right part of the map remains unstable but this is undetected by Φ ($\Phi = 0.87$, $\Phi_{\text{iw}} = 0.37$, $\Phi_{\text{pears}} = 0.22$). From (b), optimizing Φ_{pears} can cause the most frequently selected features to have highly varying importance. This is not penalized adequately by Φ_{pears} ($\Phi_{\text{pears}} = 0.57$, $\Phi_{\text{iw}} = 0.39$, $\Phi = 0.48$). Optimizing Φ_{iw} prevents these two side-effects (c) ($\Phi_{\text{iw}} = 0.63$, $\Phi = 0.92$, $\Phi_{\text{pears}} = 0.69$).

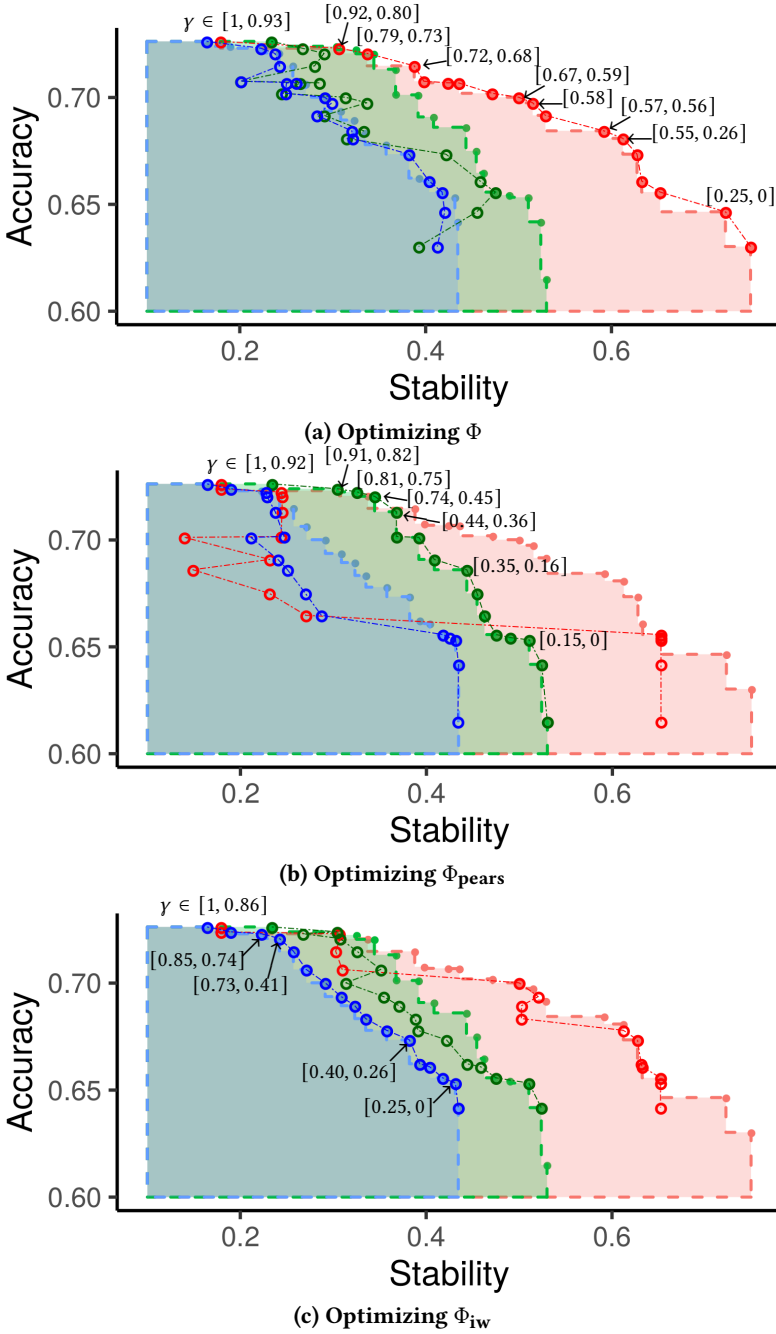


Figure B.3: Pareto-optimal curves for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on the **gravier** data set. Other example where decision is heavily affected by the stability measure. Classification performance is sacrificed in large amounts for (approx.) $\gamma \leq 0.7$ when Φ is used, and only for $\gamma \leq 0.4$ when Φ_{iw} or Φ_{pears} is considered. Accuracy is sacrificed more with Φ because it here overestimates stability.

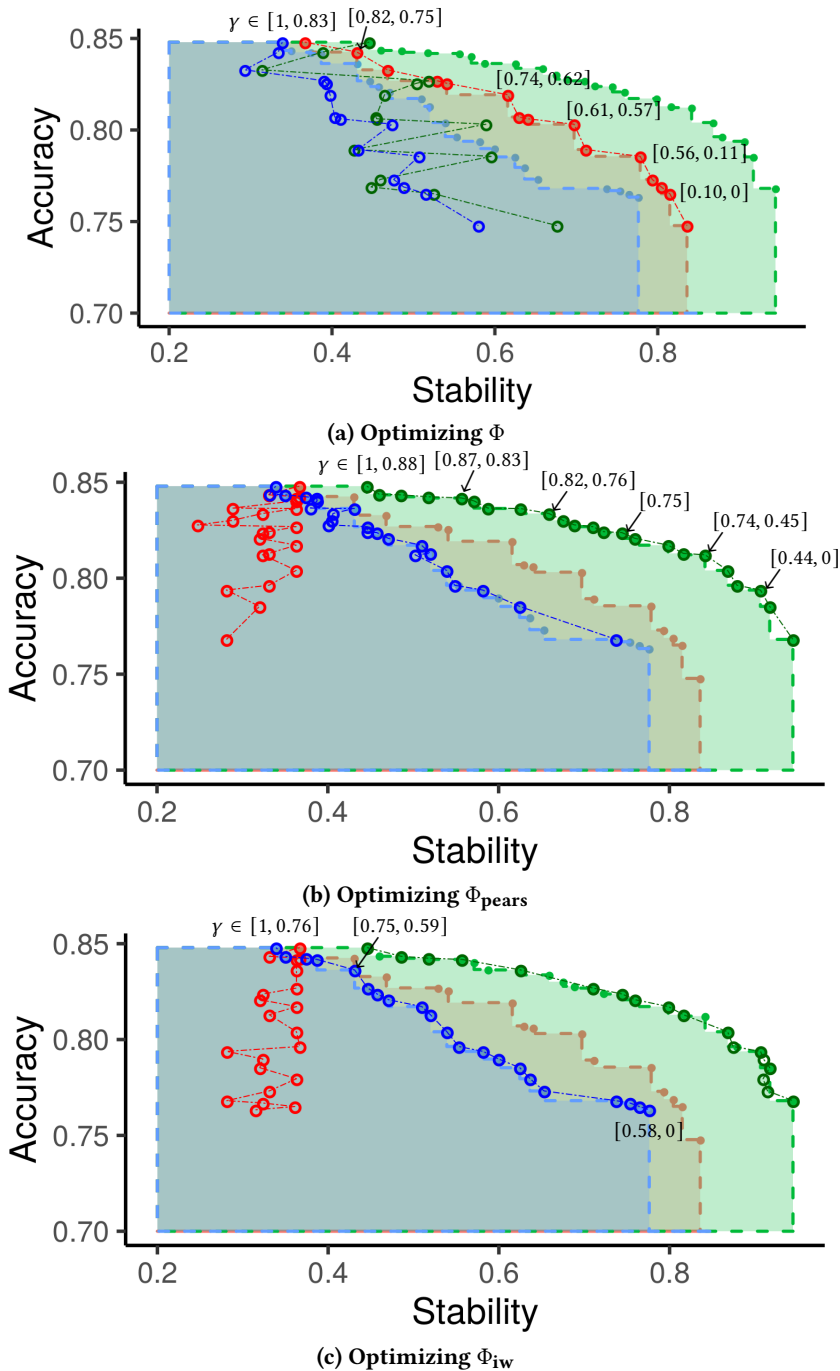


Figure B.4: Pareto-optimal curves for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on the **chiaretti** data set. The meta-parameter choices that constitute the Pareto-front are similar for Φ_{iw} and Φ_{pears} , but are vastly different for Φ .

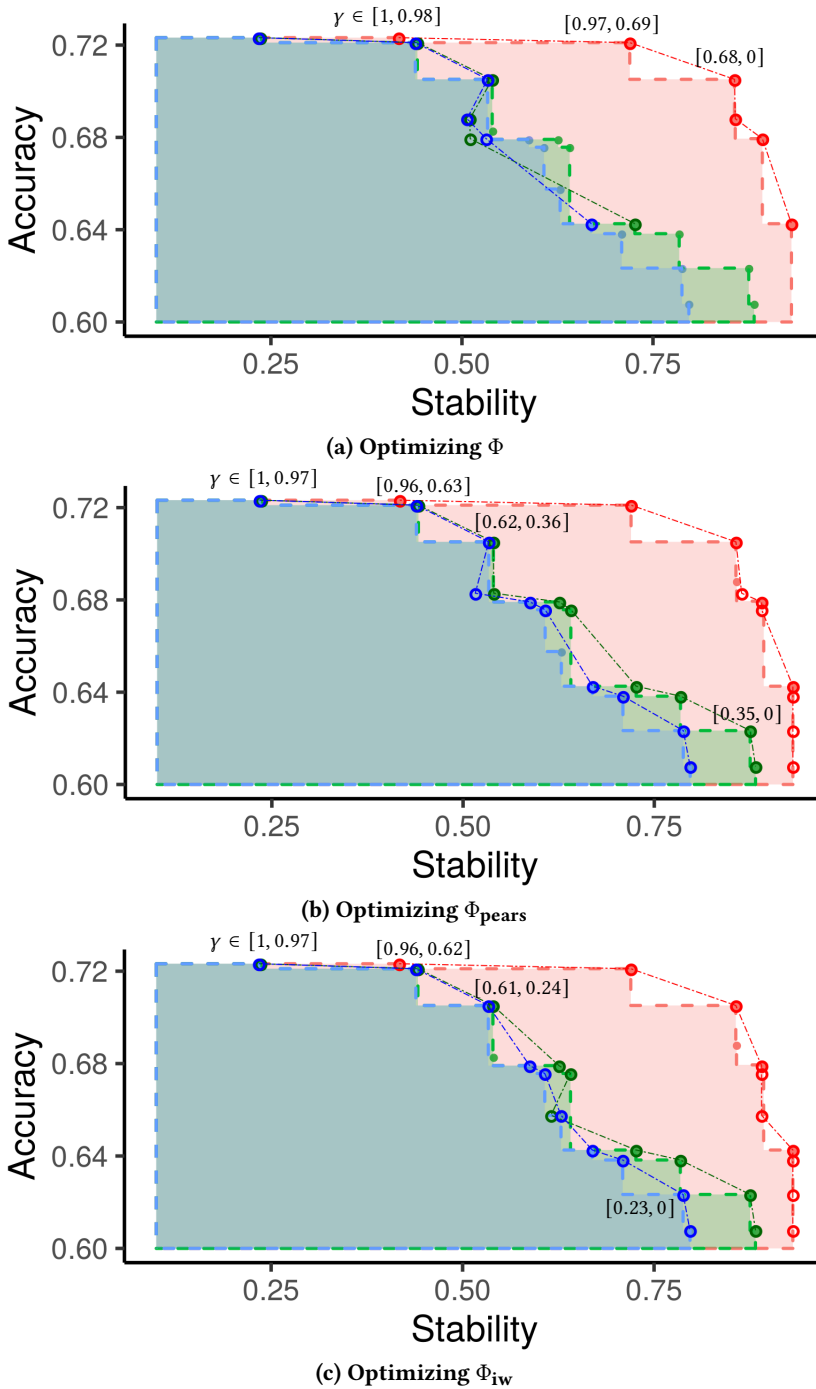


Figure B.6: Pareto-optimal curves for Φ (red), Φ_{pears} (green) and Φ_{iw} (blue), obtained with the hybrid-RFE on the *arcene* data set. The measure Φ takes values much higher than Φ_{pears} and Φ_{iw} , which indicates the stable selection of some features that are only marginally used for prediction.

Importance Weighted Stability of Standard Feature Selection Methods

C

In this Appendix, the aggregated results of Section 5.7 are depicted. Figures C.2 to C.5 compare the stability Φ_{iw} of logistic regression with a LASSO or ELASTIC NET penalty, the logistic RFE, the SVM-RFE, random forests, the RELIEF algorithm, deep feature selection without any hidden layer (shallow-dfs), with a single hidden layer of 256 units (dfs), and with three hidden layers of 256 units (deep-dfs).

Tables C.1 to C.6 depict the expected satisfactions and hypervolume of every method on each data set, as well as their per-dataset ranking. We perform below a statistical analysis of the observed differences in performance. Note that statistical significance must be treated cautiously as the data sets are from different sources. Thus, they cannot be assumed to be representative of a specific area of research.

We use the non-parametric Friedman test [Fri37] to assess the statistical significance of the differences between each algorithm's observed average rank R_j . The Friedman statistic follows a χ^2 distribution with N_a degrees of freedom

$$\chi_F^2 = \frac{12N_d}{N_a(N_a + 1)} \sum_{j=1}^{N_a} R_j^2 - \frac{(N_a)(N_a + 1)^2}{4} \quad (\text{C.1})$$

with N_a the number of algorithms being tested (here $N_a = 9$) and N_d the number of data sets (here $N_d = 11$). Results of Tables C.1 to C.6 leads to a p-value of $8.2 * 10^{-11}$. The null hypothesis of no average rank differences can be rejected with strong confidence. Standard practice is to subsequently use a post-hoc Nemenyi test [Nem63] as recommended by Demvsar [Dem06]. The resulting critical difference plot is shown in Figure C.1. There is a clear gap in performance between the best four performing approaches (the logistic-RFE, elastic-net, svm-RFE, and random forests) and the others. Deep feature selection seems to be the worst approach. This is not too surprising as deep learning is known to require large amounts of data to work properly, and

the considered data sets mostly have a low number of samples. Moreover, the stability of deep feature selection is generally the worst among the considered approaches.

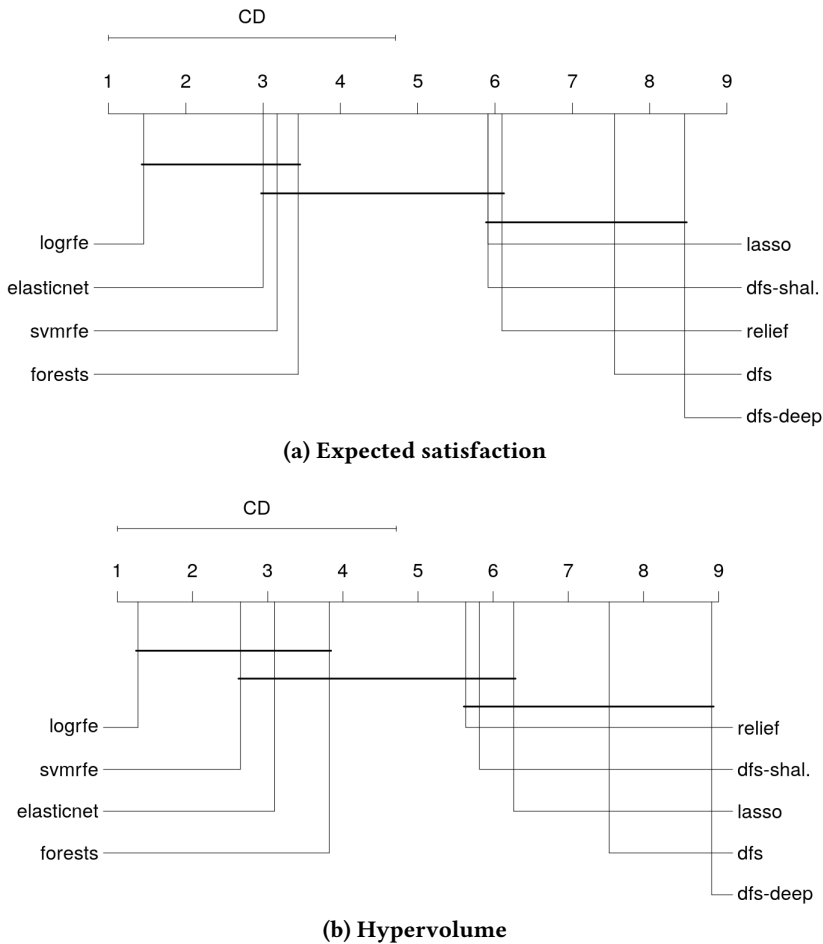


Figure C.1: Critical difference plot for the expected satisfaction (a) and the hypervolume (b). In (a), the logistic-RFE is the best overall performing algorithm, followed by the elastic-net, the svm-RFE and random forests. There is no statistically significant difference amongst these 4 algorithms. The lasso and shallow-dfs are tied in 5th place and the relief comes next, all with a statistically significant difference with the logistic-RFE only. Dfs and deep-dfs are the worst performing approaches and are significantly worse than the 4 top-performing algorithms. The ranking is the same in (b), except that the lasso, shallow-dfs and relief are re-ordered.

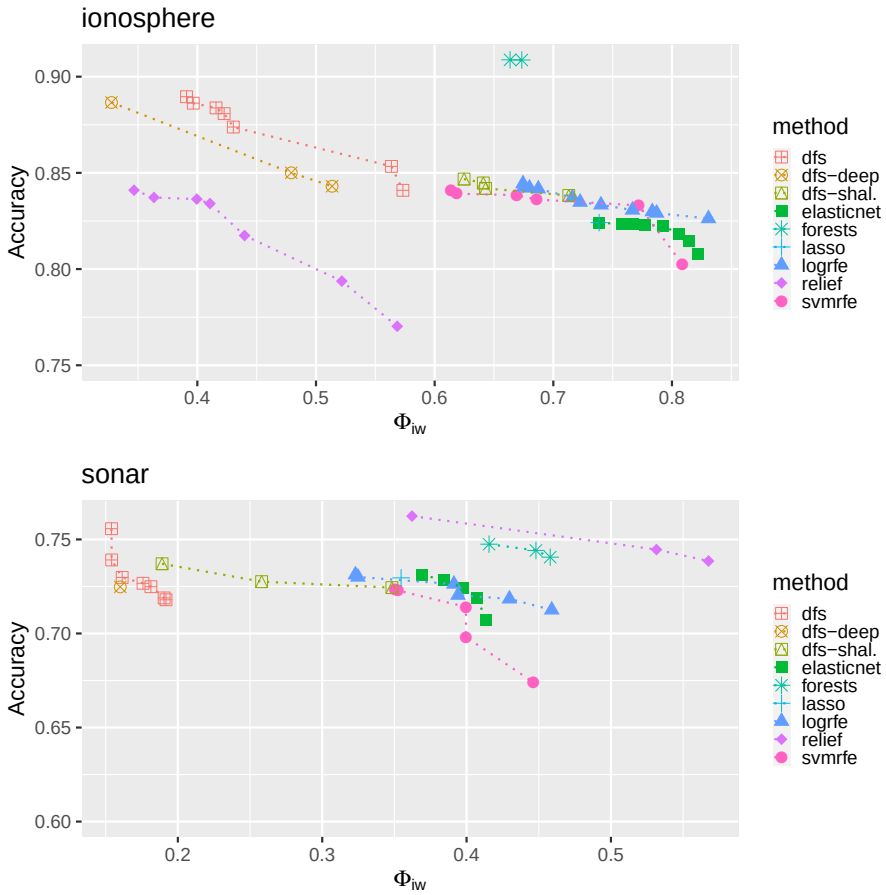


Figure C.2: Pareto-fronts (a_b, Φ_{iw}) of the considered algorithms on the **ionosphere** and **sonar** data sets. On **ionosphere**, the **ELASTIC NET**, **LASSO**, **SVM-RFE**, **LOG-RFE** have similar overall performance. The **RELIEF** reaches similar accuracies but is largely less stable. Random forests have better classification performance but are slightly less stable. Dfs and deep-dfs have stronger accuracy than the linear methods but are dominated by random forests. On **sonar**, the **RELIEF** dominates all other approaches.

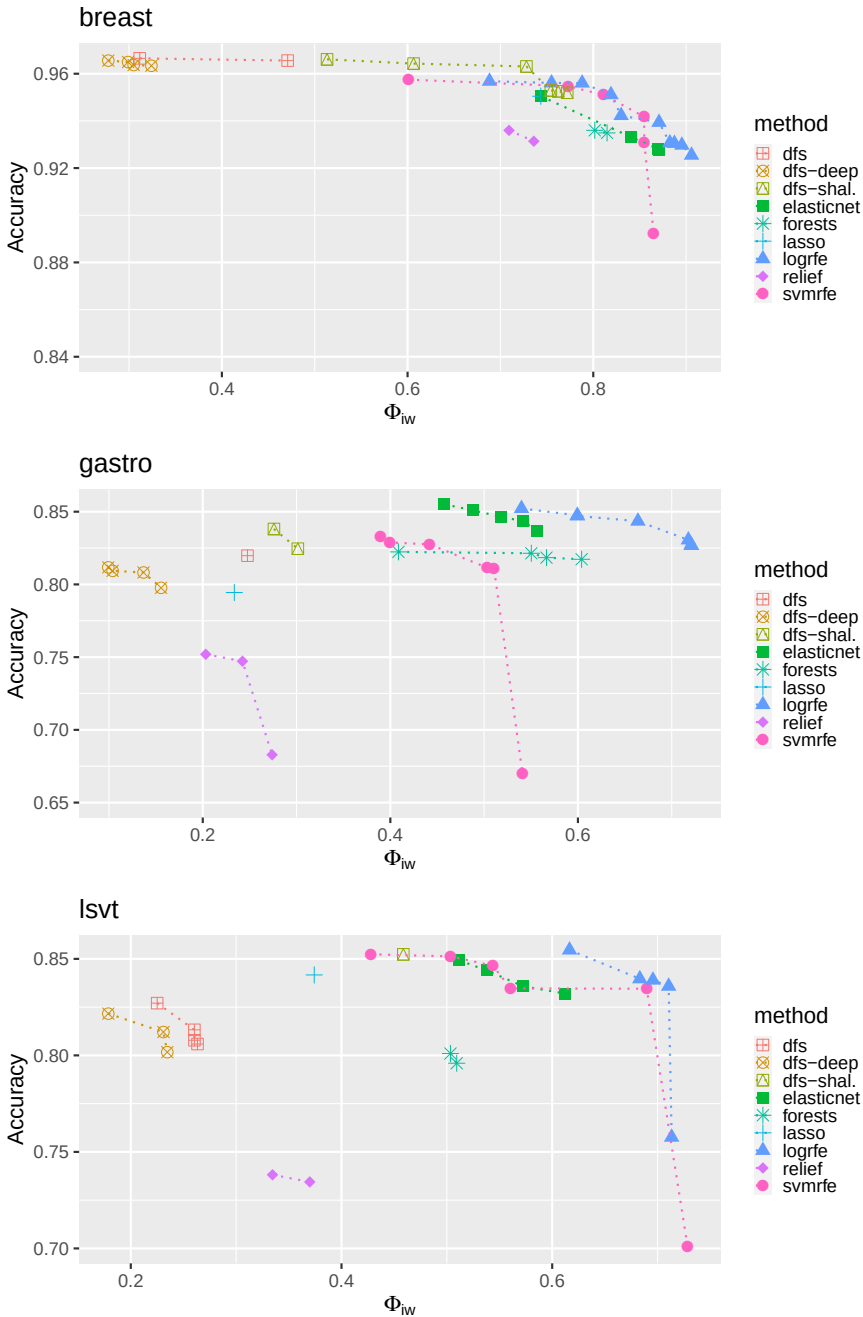


Figure C.3: Pareto-fronts (a_b, Φ_{1w}) on the **breast, **gastro** and **lsvt** data sets. The best accuracy is reached by neural networks on **breast**, but they are less stable. On **gastro** and **lsvt**, the best approaches seem to be the linear ones. The log-rfe reaches the highest stability on **gastro** and along with the svm-rfe on **lsvt**. The relief is the worst on the three data sets.**

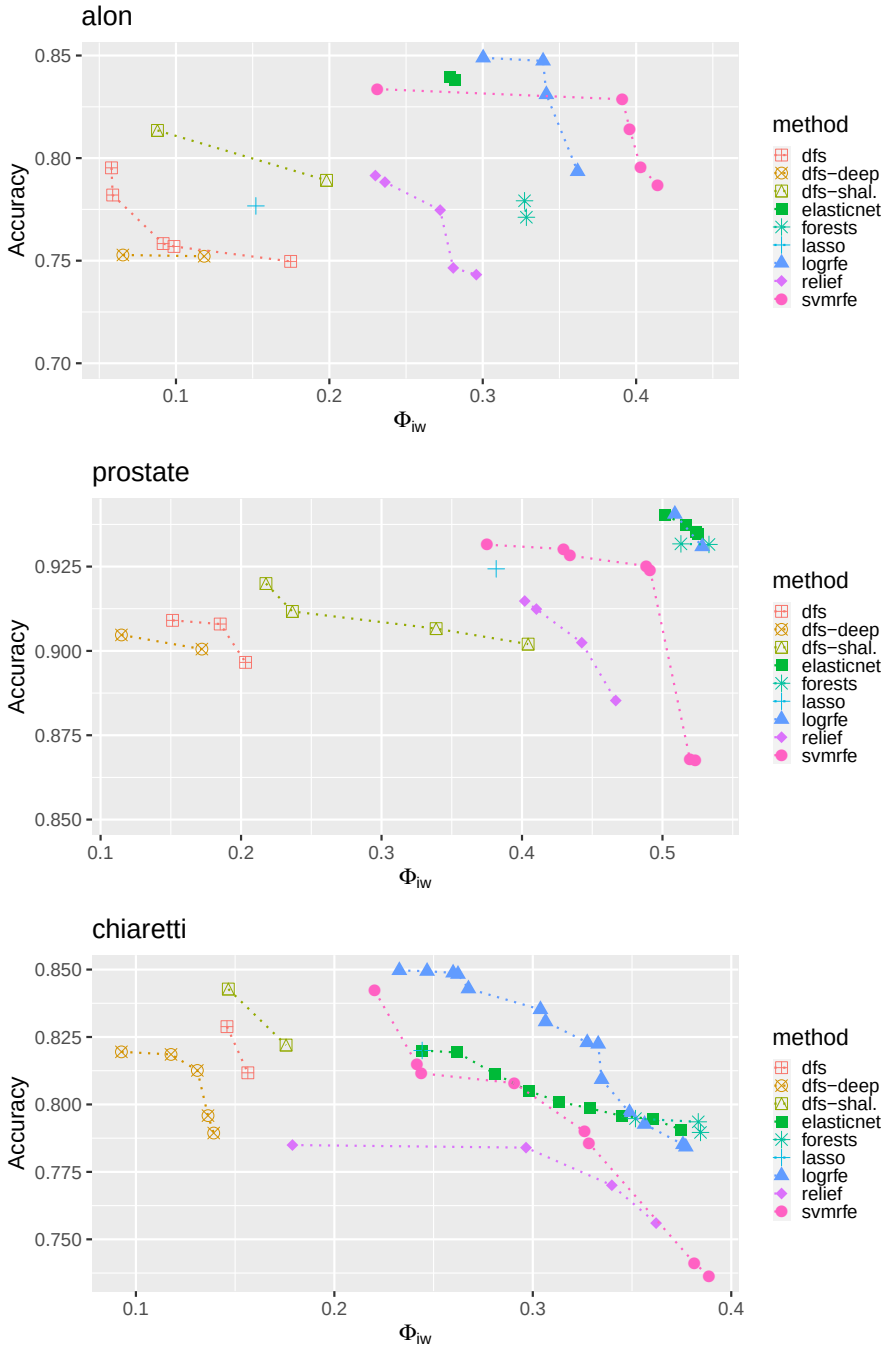


Figure C.4: Pareto-fronts (a_b, Φ_{1w}) of the considered algorithms on the **alon, **singh** and **chiaretti** data sets. Neural networks perform quite poorly on those microarray data sets, especially in terms of selection stability.**

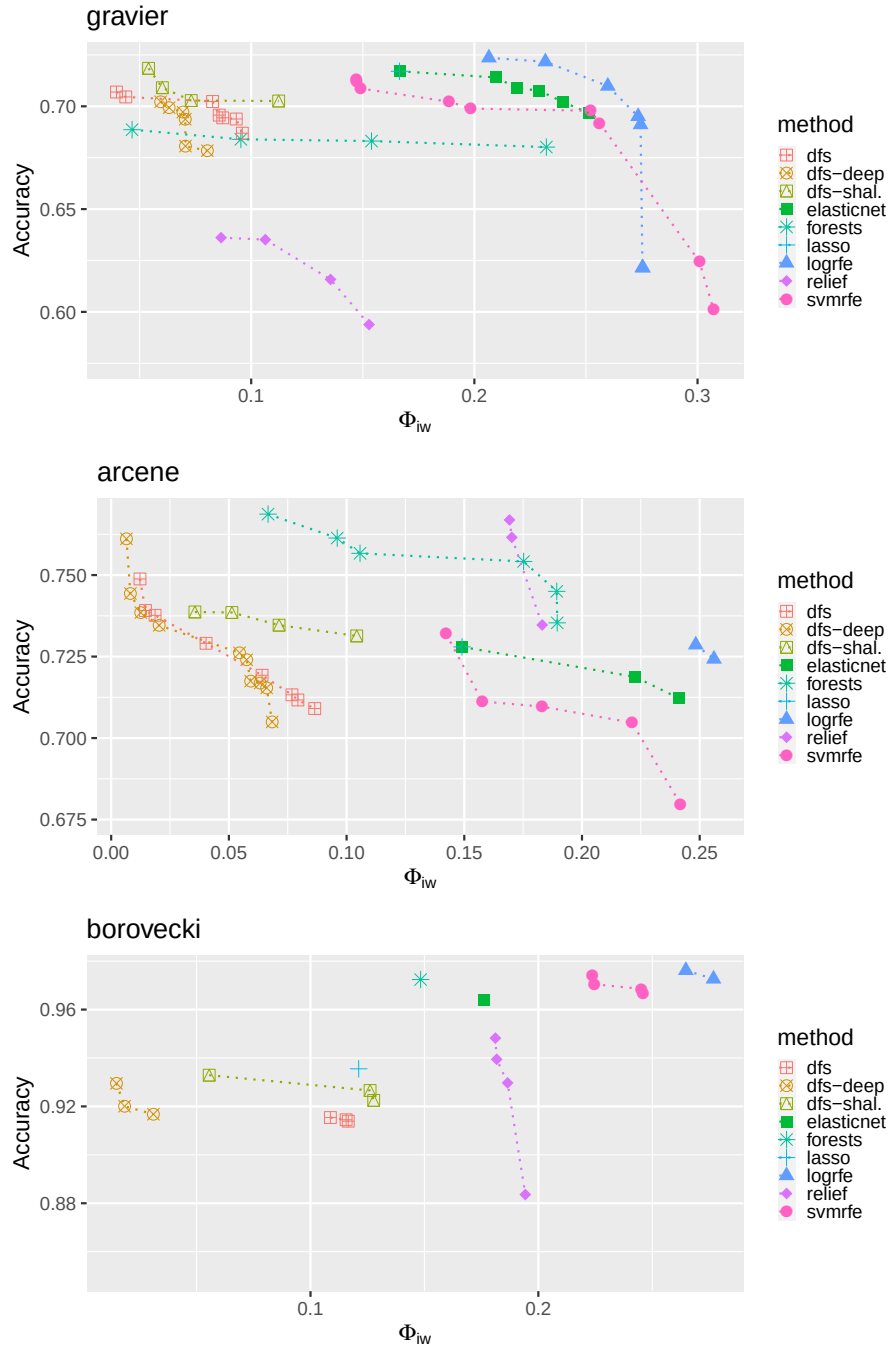


Figure C.5: Pareto-fronts (a_b, Φ_{iw}) of the considered algorithms on the **gravier**, **arcene** and **borovecki** data sets. Non linear methods perform better, at least in terms of classification performance, on **arcene**, while the linear algorithms seem to be the good choice on **gravier** and **borovecki**.

	ionosphere				sonar			
Method	E_{S_a}	rank	HV	rank	E_{S_a}	rank	HV	rank
logrfe	0.666	2	0.562	1	0.369	3	0.208	3
svmrfe	0.649	3	0.543	3	0.349	6	0.194	4
forests	0.713	1	0.550	2	0.402	2	0.225	2
elasticnet	0.643	4	0.532	4	0.361	4	0.191	5
lasso	0.621	6	0.475	6	0.350	5	0.164	6
relief	0.521	9	0.369	9	0.441	1	0.289	1
dfs-shallow	0.633	5	0.493	5	0.344	7	0.161	7
dfs	0.604	7	0.436	7	0.312	8	0.099	8
dfs-deep	0.573	8	0.384	8	0.278	9	0.076	9

Table C.1: Expected satisfaction and hypervolume of the 9 evaluated algorithms on *ionosphere* and *sonar*.

	breast				gastro			
Method	E_{S_a}	rank	HV	rank	E_{S_a}	rank	HV	rank
logrfe	0.857	1	0.819	1	0.630	1	0.505	1
svmrfe	0.848	2	0.784	2	0.521	4	0.349	4
forests	0.815	5	0.708	5	0.559	3	0.389	3
elasticnet	0.838	3	0.760	3	0.573	2	0.392	2
lasso	0.803	6	0.669	6	0.384	7	0.140	7
relief	0.775	7	0.639	7	0.334	9	0.135	8
dfs-shallow	0.824	4	0.718	4	0.455	5	0.207	5
dfs	0.693	8	0.443	8	0.419	6	0.158	6
dfs-deep	0.620	9	0.304	9	0.365	8	0.098	9

Table C.2: Expected satisfaction and hypervolume of the 9 evaluated algorithms on *breast* and *gastro*.

	lsvt				alon			
Method	E_{S_a}	rank	HV	rank	E_{S_a}	rank	HV	rank
logrfe	0.633	1	0.501	1	0.489	1	0.253	2
svmrfe	0.620	2	0.496	2	0.488	2	0.277	1
forests	0.496	5	0.304	5	0.401	4	0.182	4
elasticnet	0.587	3	0.426	3	0.455	3	0.195	3
lasso	0.496	6	0.257	6	0.333	8	0.088	8
relief	0.362	9	0.174	7	0.385	5	0.170	5
dfs-shallow	0.545	4	0.321	4	0.374	6	0.119	6
dfs	0.424	7	0.174	8	0.336	7	0.096	7
dfs-deep	0.407	8	0.152	9	0.294	9	0.060	9

Table C.3: Expected satisfaction and hypervolume of the 9 evaluated algorithms on **lsvt and **alon**.**

	singh				chiaretti			
Method	E_{S_a}	rank	HV	rank	E_{S_a}	rank	HV	rank
logrfe	0.682	1	0.462	1	0.472	1	0.256	1
svmrfe	0.652	4	0.448	4	0.448	2	0.245	2
forests	0.679	3	0.462	3	0.441	4	0.228	4
elasticnet	0.682	2	0.462	2	0.446	3	0.236	3
lasso	0.600	6	0.328	7	0.418	5	0.158	6
relief	0.608	5	0.384	5	0.403	7	0.206	5
dfs-shallow	0.587	7	0.336	6	0.407	6	0.121	7
dfs	0.494	8	0.170	8	0.388	8	0.104	8
dfs-deep	0.478	9	0.144	9	0.368	9	0.088	9

Table C.4: Expected satisfaction and hypervolume of the 9 evaluated algorithms on **singh and **chiaretti**.**

Method	gravier				arcene			
	E_{S_a}	rank	HV	rank	E_{S_a}	rank	HV	rank
logrfe	0.302	1	0.122	1	0.313	3	0.118	1
svmrfe	0.282	3	0.119	2	0.291	5	0.109	3
forests	0.251	5	0.087	4	0.322	2	0.098	5
elasticnet	0.287	2	0.110	3	0.297	4	0.111	2
lasso	0.272	4	0.073	5	0.278	6	0.068	6
relief	0.164	9	0.04	8	0.329	1	0.100	4
dfs-shallow	0.246	6	0.05	6	0.269	8	0.051	7
dfs	0.231	7	0.04	7	0.262	9	0.041	8
dfs-deep	0.222	8	0.035	9	0.269	7	0.032	9

Table C.5: Expected satisfaction and hypervolume of the 9 evaluated algorithms on *gravier* and *arcene*.

Method	borovecki			
	E_{S_a}	rank	HV	rank
logrfe	0.609	1	0.264	1
svmrfe	0.589	2	0.237	2
forests	0.544	4	0.140	5
elasticnet	0.549	3	0.165	4
lasso	0.492	6	0.112	6
relief	0.534	5	0.176	3
dfs-shallow	0.486	7	0.111	7
dfs	0.468	8	0.099	8
dfs-deep	0.439	9	0.034	9

Table C.6: Expected satisfaction and hypervolume of the 9 evaluated algorithms on *borovecki*.

Correction for chance in correlated spaces and the impact of the similarity matrix S

D

This appendix studies the property of correction for chance in the presence of feature correlations. We show that correcting for chance can be, in some cases, orthogonal to the objective of addressing stability underestimation in correlated spaces. This effect is highly linked to the density of the similarity matrix S . It should not be overlooked when opting for a specific choice of similarity matrix. Furthermore, there is a direct conflict between the correction for chance and the lower bound properties. Current correlation-aware measures are all either lower-bounded or corrected for chance, but not both. In the last part of this appendix, we derive a strictly lower-bounded and asymptotically corrected for chance stability measure. Nonetheless, the computational cost of this approach is admittedly prohibitive, and simply sparsifying S to a pre-defined threshold might be the best overall option.

D.1 Context

As detailed in Section 5.3.3, a good property of a stability measure is its correction for chance, *i.e.*, the expected value of the measure should be 0 whenever features are selected uniformly at random. The importance-weighted measure Φ_{iw} provably satisfies this property. The correlation aware measure Φ_{msi} does not, however. This appendix analyses this issue and studies different solutions.

When ignoring feature correlations, the basic principle of correction for chance is that a stability measure should **not** be influenced by feature intersections that occur by chance rather than by the decision-making of the feature selection algorithm. Non-corrected stability measures are biased towards the selection of large feature sets. Such large feature sets have a higher probability of containing identical features by chance. For instance, consider the following scenario. Method A has an average feature intersection of 3 when selecting 5 features out of 10. Method B has an average feature intersection

of 5 when selecting 7 features out of 10. To measure selection stability, one could opt for the Dice-Sørensen Index, as done in [YDL08] as a comparison point to their new measure. This index is defined as

$$\text{Dice} = \frac{2}{M(M-1)} \sum_{i=1}^M \sum_{j=i+1}^M \frac{2|\mathcal{F}_i \cap \mathcal{F}_j|}{k_i + k_j}. \quad (\text{D.1})$$

It is not corrected for chance. In particular, when a constant number of features k is selected in each run, $E[\text{Dice}]|H_0 = \frac{k}{d} > 0$. We have here

$$\text{Dice}(A) = \frac{3}{5} < \text{Dice}(B) = \frac{5}{7}.$$

Yet, as method B selects 7 features out of only 10, the probability that it selects identical variables in different runs by chance is very high. Using the corrected for chance Kuncheva Index, we get

$$KI(A) = \frac{3 - \frac{25}{10}}{5 - \frac{25}{10}} = 0.2 > KI(B) = \frac{5 - \frac{49}{10}}{7 - \frac{49}{10}} = 0.048.$$

Method B has a higher Dice Index, but its Kuncheva Index is close to 0, which indicates that its average feature intersection is barely above the one obtained with the random selection strategy. Classical correction for chance only depends on the number of selected features at each run k_i (here constant and equal to k) and the total number of input features d . Correcting for chance also allows a sound comparison of stability values measured on different data sets, as they might all have a different number of input features d .

Correcting for chance in the presence of feature correlations is not that straightforward, as the expected value of stability when features are selected randomly depends on the whole similarity matrix $\mathcal{S}$. The correction for chance thus depends on the similarity between irrelevant features. Such features might never be selected, but they influence the value of any corrected for chance stability measure. This causes several problems detailed in Sections D.2 and D.3. In particular, no matter how stable the selection appears to be, no matter how close Φ , Φ_{iw} and Φ_{msi} are to 1 (but not equal), a corrected for chance measure could still take unbounded negative values. This illustrates that special caution must be taken when one deals with correction for chance in correlated spaces.

To show that, we first introduce the following corrected for chance variant of our measure:

$$\Phi_{\text{msi}}^{\text{cfc}} = \frac{\Phi_{\text{msi}} - \Phi_{\text{msi}}|H_0}{1 - \Phi_{\text{msi}}|H_0} \quad (\text{D.2})$$

As estimating Φ_{msi} requires to solve a constrained optimization problem, there is no close formula for $\Phi_{\text{msi}}|H_0$. To evaluate $\Phi_{\text{msi}}|H_0$, we simply randomize the selected feature identities in the M runs and compute Φ_{msi} for this sampling. As doing so is inherently stochastic, we do this $B = 10$ times and report the average stability. The variant $\Phi_{\text{msi}}^{\text{cfc}}$ is corrected for chance but is stochastic in nature. Furthermore, as $\Phi_{\text{msi}}|H_0$ can be as close to 1 as desired (by tuning the similarity matrix $\mathcal{S}$), $\Phi_{\text{msi}}^{\text{cfc}}$ looses the lower bound property. Zhang et al. opted for a similar strategy to correct the POGR measure for chance [Zha+09]. The resulting stability measure, denoted nPOGR, naturally loses the lower bound property as well.¹

The initial objective of introducing stability measures that consider feature correlations is to address the stability *underestimation* in correlated spaces. Sechidis et al. notably write

The particular contribution of this paper is on how to estimate stability in the presence of *correlated features*, also known as *feature redundancy*. We will demonstrate that any stability measure not taking such redundancy into account necessarily gives a *systematic under-estimate* of the stability, thus giving an overly pessimistic view of a given FS algorithm. This systematic underestimation of stability can have a variety of consequences, depending on the application domain.

They indeed observe such a stability underestimation in their paper [Sec+19]. Zhang et al. reach the same conclusion, noting that the corrected for chance measure nPOGR may be more appropriate:

Based on microarray data sets for three diseases, we show that though there are few genes shared between the DEG lists from different studies for a disease, most of the POGR and nPOGR scores are rather high, suggesting high reproducibility of the DEG lists in the sense that the apparently inconsistent DEG lists are significantly correlated. [...] After excluding the effects of list lengths and general data correlations on chance agreement, the nPOGR score may be more appropriate for comparing the consistency levels of DEG lists with different lengths and from different studies. On the other hand, because the nPOGR score measures the relative agreement beyond chance, it should be necessary to use the POGR score to reflect the apparent agreement of DEG lists based on the original data correlation.

¹Strictly speaking, the nPOG measure does satisfy the lower bound property as it is defined as the maximum of 0 and its original value. However, this is an unsatisfactory fix. The issue is that the measure can collapse in some cases studied later in this appendix. Whether it collapses to 0 or to $-\infty$ has little importance.

The classic scenario where stability is largely increased by factoring in feature correlation has a clear interpretation: the identities of the selected features are unstable, but the algorithm tends to alternate between highly similar features. The information conveyed by the selected features is, in fact, stable. Sechidis et al. and Zhang et al. both observe this phenomenon. However, as the measures Φ_S and $nPOGR$ are not lower bounded to the contrary of Φ and $nPOG$, the opposite must be possible. The interpretation of a situation where, for instance, $nPOG \gg nPOGR$ or $\Phi_{iw} \gg \Phi_{msi}^{cfc}$ is more challenging. The selected features are stable themselves, yet the information behind them is deemed unstable. We show further below that such situations can occur with correlated for chance measures, especially (but not only) when the similarity matrix is dense. Sechidis et al. avoids this phenomenon by posing $s_{f,g} = \delta(|\rho_{f,f'}| > 0.5)$ which greatly reduces the density of S . Yet, by doing so, they 1) abandon the capability of their measure to use fractional similarity values and 2) drop a large fraction of the small but existing feature correlations. The specific threshold value, here set to 0.5, also largely influences stability. Zhang et al. rather use statistical analysis to find significantly correlated features. This essentially comes back to posing $s_{f,g} = \delta(|\rho_{f,f'}| > \theta)$, with θ such that the p-value of a feature pair with $|\rho_{f,f'}| > \theta$ is below the considered significance threshold (typically 0.05).

In order to study in more detail the hidden issues of correction for chance, we analyze the behavior of Φ_{msi} and Φ_{msi}^{cfc} in different scenarios (including with different similarity matrices). We reflect on the advantages and disadvantages of both variants.

D.2 Simulated data: revisiting scenario 3

This section reuses the third scenario of the simulation studies of sections 6.3.2 and 6.4.4 (see Table 6.2), with the following modification. The irrelevant variables are now correlated to each other, and this correlation amounts to ρ^{oog} on average. We set the size q of the 5 relevant correlated groups to 5, and we study stability in function of ρ^{oog} in Figure D.1a. When $\rho^{oog} = 0$, Φ , Φ_{iw} and Φ_{msi} are the same as in Figure 6.6c when $q = 5$. The measures Φ and Φ_{iw} are not so good because the lasso picks only a single or a few features inside each group, which need not be the same in different selection runs. The stability Φ_{msi} deals with feature correlations and provides a more accurate stability value. It is quite good here because the five groups are selected in most of the runs, with only a few irrelevant features that change from run to run. The correction for chance is negligible as most features (the 1000 irrelevant ones) are not correlated to any other features. When ρ^{oog} increases, Φ , Φ_{iw} and Φ_{msi} do not vary much. The measure Φ_{msi}^{cfc} does not behave like the others and decreases towards large negative values. It does so because

$\Phi_{\text{msi}}|H_0$ follows closely ρ^{og} and tends to 1. We here argue that such a behavior can be quite detrimental to the interpretability of stability. Should not an algorithm that seems to consistently select features from the same correlated groups be considered stable, whether or not irrelevant features are correlated to each other? This is especially true since some classical stability is observed ($\Phi, \Phi_{\text{iw}} \approx 0.3 - 0.4$). The measure Φ_{msi} is not satisfactory either as $\Phi_{\text{msi}}|H_0$ reaches high values. A large value for Φ_{msi} is thus not a trustable indicator of stability occurring via the decision-making of the selection algorithm.

Figure D.1a has been obtained with $s_{f,g} = |\rho_{f,g}|$ with ρ denoting the Spearman's correlation. In that case, the similarity matrix S is dense. Nonetheless, the same issue happens when thresholding the pairwise correlation. Figure D.1b is obtained by setting all similarities below $\theta = 0.5$ to 0. The same issue arises as ρ^{og} eventually becomes greater than θ . When $\rho^{\text{og}} < \theta$, correction for chance is essentially non-existent and the $\Phi_{\text{msi}}, \Phi_{\text{msi}}^{\text{cfc}}$ and $\Phi_{\text{msi}}|H_0$ curves remain flat. When $\rho^{\text{og}} > \theta$, stability values quickly reach the same values as in Figure D.1a. Varying θ simply moves the crossover point. Naturally, $\Phi_{\text{msi}}^{\text{cfc}}$ can be prevented from reaching negative values by setting θ to 1, but then $\Phi_{\text{msi}}^{\text{cfc}}$ becomes equivalent to Φ_{iw} , and correlation between features is simply ignored. Using $s_{f,g} = \delta(|\rho_{f,f'}| > 0.5)$, as in [Sec+19], simply makes the crossover point harsher (Figure D.1c). Note that Figure D.1 is obtained while keeping $\rho^{\text{og}} \leq 0.95$ for visualization reasons, as $\Phi_{\text{msi}}^{\text{cfc}} \approx -3$ when $\rho^{\text{og}} = 1$.

In conclusion, neither Φ_{msi} nor $\Phi_{\text{msi}}^{\text{cfc}}$ are truly satisfactory in the studied scenario. An ideal measure would have its expected value under H_0 stay at 0 for all ρ^{og} values, *i.e.*, it should be corrected for chance. It should also never decrease towards arbitrary large negative values. Note that such behavior can naturally be obtained by simply taking the maximum between 0 and $\Phi_{\text{msi}}^{\text{cfc}}$, but this feels more like a trick to hide the underlying issue of the measure collapsing. The same is true for $\max(\Phi_{\text{msi}}^{\text{cfc}}, \Phi)$ or $\max(\Phi_{\text{msi}}^{\text{cfc}}, \Phi_{\text{iw}})$, *i.e.*, the collapse still occurs, just to a more limited extent.

D.3 Stability on microarray data sets

Consider the logistic RFE algorithm with the same set of meta-parameters as in Section 6.5, and applied to the a1on microarray data set (we here set M to 30). Stability according to Φ and Φ_{iw} is quite poor but pretty average when compared to the other microarray data sets under study (0.34-0.29). The measure Φ_{msi} is much higher (0.71). However, it turns out that $\Phi_{\text{msi}}|H_0 = 0.67$. As a consequence, $\Phi_{\text{msi}}^{\text{cfc}}$ is equal to $\frac{0.71-0.67}{1-0.67} \approx 0.096$, which is really low². Figure D.2 details two correlation plots obtained on the a1on data set. These correlation plots are computed based on two specific selection runs. For clarity, features

²It is even possible to find meta-parameters sets for which $\Phi_{\text{msi}}^{\text{cfc}} = -0.1$

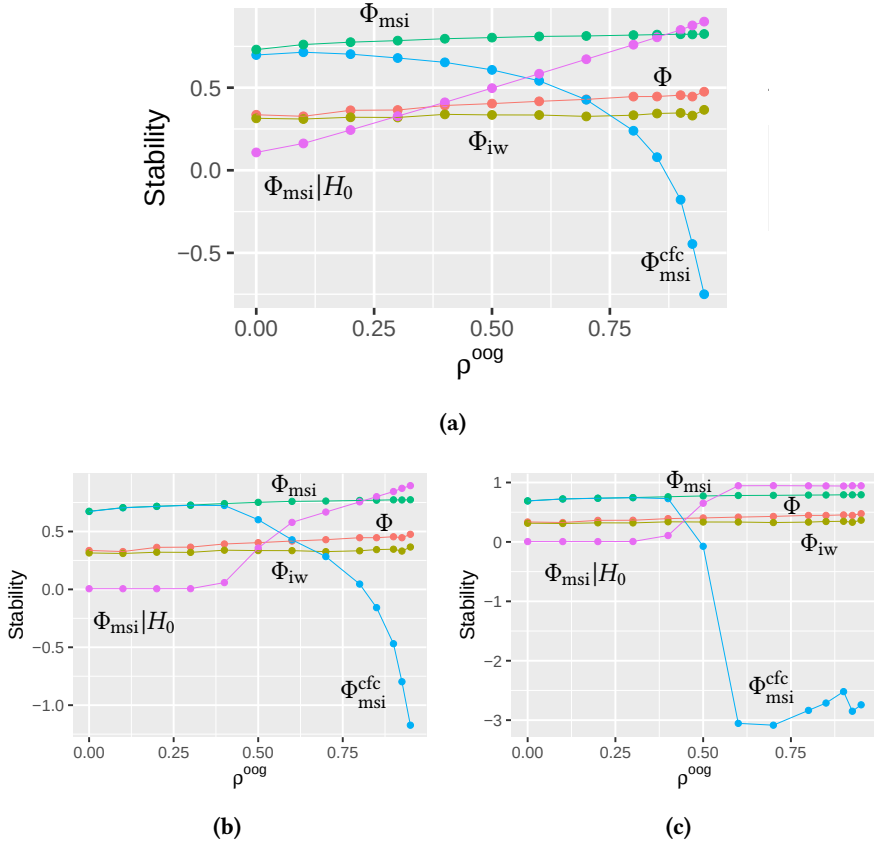


Figure D.1: Stability dependency on the correlation ρ^{oog} between irrelevant features in the revisited scenario 3 from Table 6.2. The corrected for chance variant $\Phi_{\text{msi}}^{\text{cfc}}$ is unsatisfactory as it tends towards large negative values when ρ^{oog} increases. The base variant Φ_{msi} is also not satisfactory as its expected value $\Phi_{\text{msi}}|H_0$ reaches high values. The similarities $s_{f,g}$ are set to the absolute value of the Spearman's correlation $|\rho_{f,g}|$ in (a). Values lower than $\theta = 0.5$ are set to 0 in (b) while values higher than $\theta = 0.5$ are subsequently set to 1 in (c).

are sorted by solving the corresponding bipartite matching problem. The first one (Figure D.2a) is obtained via the logistic-RFE algorithm. The second one (Figure D.2b) comes from random selection. Both plots have approximately the same stability, according to Φ_{msi} (≈ 0.7). Yet, they are completely different. In Figure D.2a, nine features are selected in both runs and have a maximal correlation with each other. The bottom right part of the plot shows that most of the remaining variables contribute only slightly to stability: they are different from each other and only slightly correlated towards their respective matched variables. Still, 9/20 features are selected in both runs, which seems somewhat stable at the very least.

The second plot (Figure D.2b), obtained with random selection, is much more homogeneous. No feature is selected in both runs, but it is possible to find a matching where every feature contributes significantly to stability because the alon data set contains a lot of correlated variables. It turns out that both graphs have similar stability according to Φ_{msi} . In this context, the corrected for chance variant $\Phi_{\text{msi}}^{\text{cfc}}$ is close to 0 despite some evident stability in Figure D.2a. Correcting for chance here would make us conclude that the algorithm is highly unstable and untrustworthy even though it is not. It just turns out that many irrelevant and never selected variables are highly correlated to each other. Figure D.3 depicts the stability values obtained with the logistic-RFE on the 6 considered data sets, according to Φ , Φ_{iw} , Φ_{msi} and $\Phi_{\text{msi}}^{\text{cfc}}$. No real increase in stability is observed with $\Phi_{\text{msi}}^{\text{cfc}}$, unless for borovecki (-0.19 on alon, -0.06 on arcene, +0.43 on borovecki, ± 0 on chiaraetti, +0.03 on gravier +0.05 on singh, compared to Φ_{iw}). This quite contradicts the findings of Zhang et. al. [Zha+09].

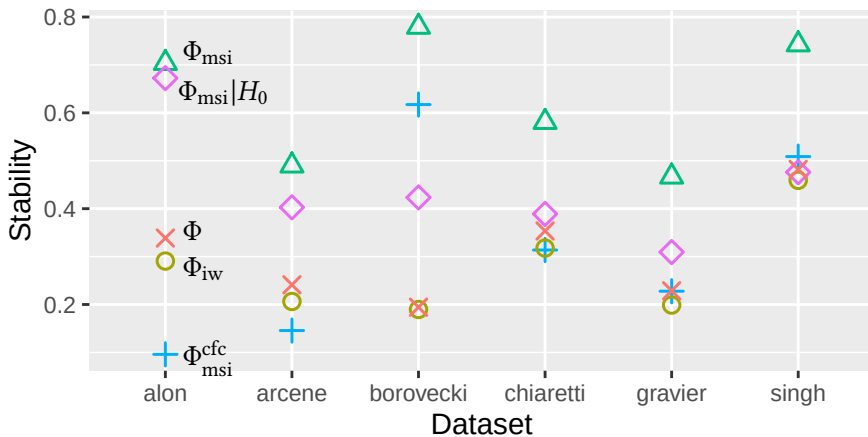
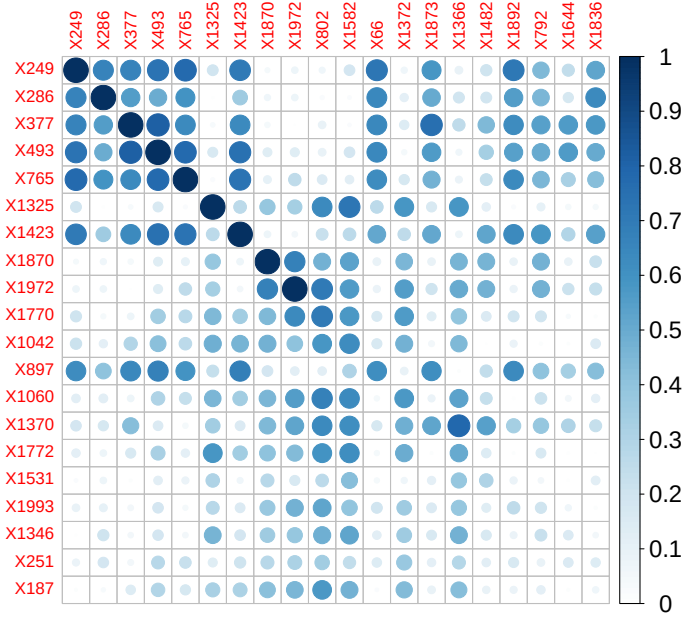
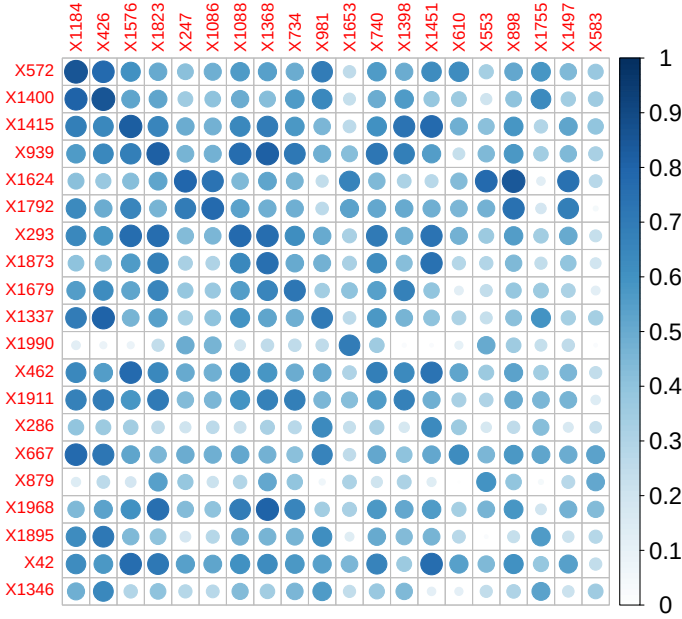


Figure D.3: The different stabilities of the logistic-RFE algorithm on 6 data sets.



(a) logrfe



(b) random selection

Figure D.2: Correlation plots between the selected features obtained with the logistic-RFE (a) and random selection (b) in two specific selection runs. Both have nearly identical stability according to Φ_{msi} . The corrected for chance measure $\Phi_{\text{msi}}^{\text{cfc}}$ is close to zero for the plot obtained through the logistic-RFE, despite some evident stability (9/20 features selected in both runs).

This issue of overly brutal correction for chance can be greatly mitigated by sparsifying the similarity matrix S , which reduces the impact of the correction for chance. We here use a different thresholding mechanism than Sechidis et. al. We pose

$$s_{f,g} = \begin{cases} |\rho_{f,g}| & \text{if } \min(\text{ranks}(|\rho_{f,-}|)[g], \text{ranks}(|\rho_{-,g}|)[f]) \leq K \\ 0 & \text{otherwise} \end{cases} \quad (\text{D.3})$$

with, for instance, $K = 0.2\frac{d}{k}$. The similarity matrix S is sparsified *locally*, instead of globally as in [Sec+19; Zha+09]. The K highest correlations are kept for each variable. With this value of K , a randomly selected feature in a run m_1 has a probability³ of $\approx \frac{1}{5}$ to have a correlated feature selected in another run m_2 . This should already decrease $\Phi_{\text{msi}}|H_0$ considerably without completely killing the signal. Figure D.4 shows the values taken by Φ_{msi} and $\Phi_{\text{msi}}^{\text{cfc}}$ using this sparsification mechanism. The measure $\Phi_{\text{msi}}^{\text{cfc}}$ is now systematically above Φ_{iw} , which is much more consistent with [Zha+09; Sec+19] (+0.07 on alon, +0.04 on arcene, +0.52 on borovecki, + 0.12 on chiaretti, +0.07 on gravier +0.20 on singh, compared to Φ_{iw}). The contrast between Figures D.3 and D.4 naturally indicates that Φ_{msi} and $\Phi_{\text{msi}}^{\text{cfc}}$ depend a lot on the level of sparsification K applied on the similarity matrix S . If K is set to a too low value, feature correlations are simply ignored and $\Phi_{\text{msi}}^{\text{cfc}} = \Phi_{\text{iw}}$. If K is too large, the correlation for chance can interfere with sound stability analysis. Figure D.5 depicts this dependency of the different stability measures on K . Increasing K above 1 systematically increases stability, as feature selection algorithms (here the logistic-RFE) often select different but highly correlated features across selection runs. Nonetheless, when K increases further, $\Phi_{\text{msi}}|H_0$ rises a lot such that $\Phi_{\text{msi}}^{\text{cfc}}$ decreases, sometimes even below Φ_{iw} (dotted lines).

³assuming $\frac{k}{d} \rightarrow 0$ such that probabilities can simply be multiplied.

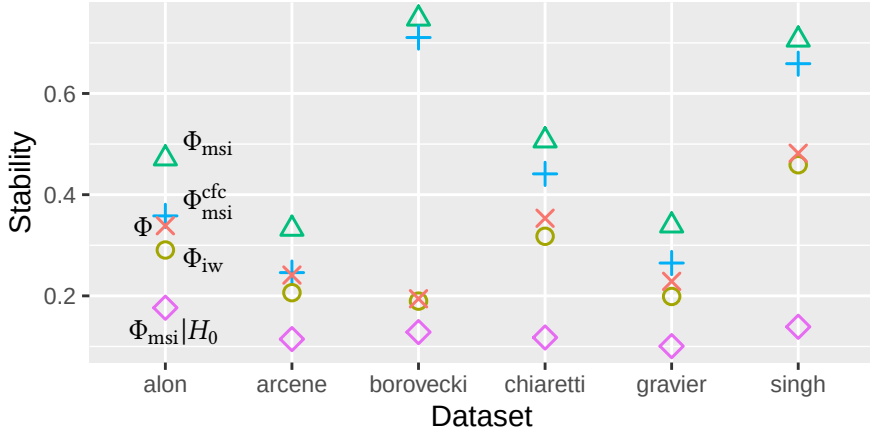


Figure D.4: The different stabilities of the logistic-RFE algorithm on 6 data sets when the similarity matrix S is sparsified according to Equation D.3 with $K = 0.2 \frac{d}{k} = \frac{d}{100}$.

One way to remove this dependency on K is to simply define stability as the maximum of $\Phi_{\text{msi}}^{\text{cfc}}$ for any K :

$$\Phi_{\text{msi}}^{\text{max}} \triangleq \max_{K \in [1, d]} \Phi_{\text{msi}}^{\text{cfc}}(K). \quad (\text{D.4})$$

As $\Phi_{\text{msi}}^{\text{max}}$ is provably equivalent to Φ_{iw} for $K = 1$, it satisfies the lower-bound property (as Φ_{iw} does). It is not corrected for chance in the strict sense as the maximum of random variables with expected values of 0 has itself an expected value above 0. Nonetheless, $\Phi_{\text{msi}}^{\text{max}}$ is asymptotically corrected for chance as $M \rightarrow \infty$. Indeed, when $M \rightarrow \infty$, the variance of $\Phi_{\text{msi}}^{\text{cfc}}(K)$ tends toward 0 and so does $\Phi_{\text{msi}}^{\text{max}}$. Figure D.6 shows that $\Phi_{\text{msi}}^{\text{max}}$ does not suffer from the same collapse as $\Phi_{\text{msi}}^{\text{cfc}}$ in the simulated scenario of Figure D.1. Note that Φ_{msi} and $\Phi_{\text{msi}}^{\text{max}}$ behave equivalently in this idealised scenario. This is naturally not always the case as highlighted previously on microarray data (e.g., Figure D.5 where $\Phi_{\text{msi}}^{\text{max}}$, the maximum of the green curve, often largely differs from Φ_{msi} with a dense similarity matrix ($K \rightarrow \infty$)).

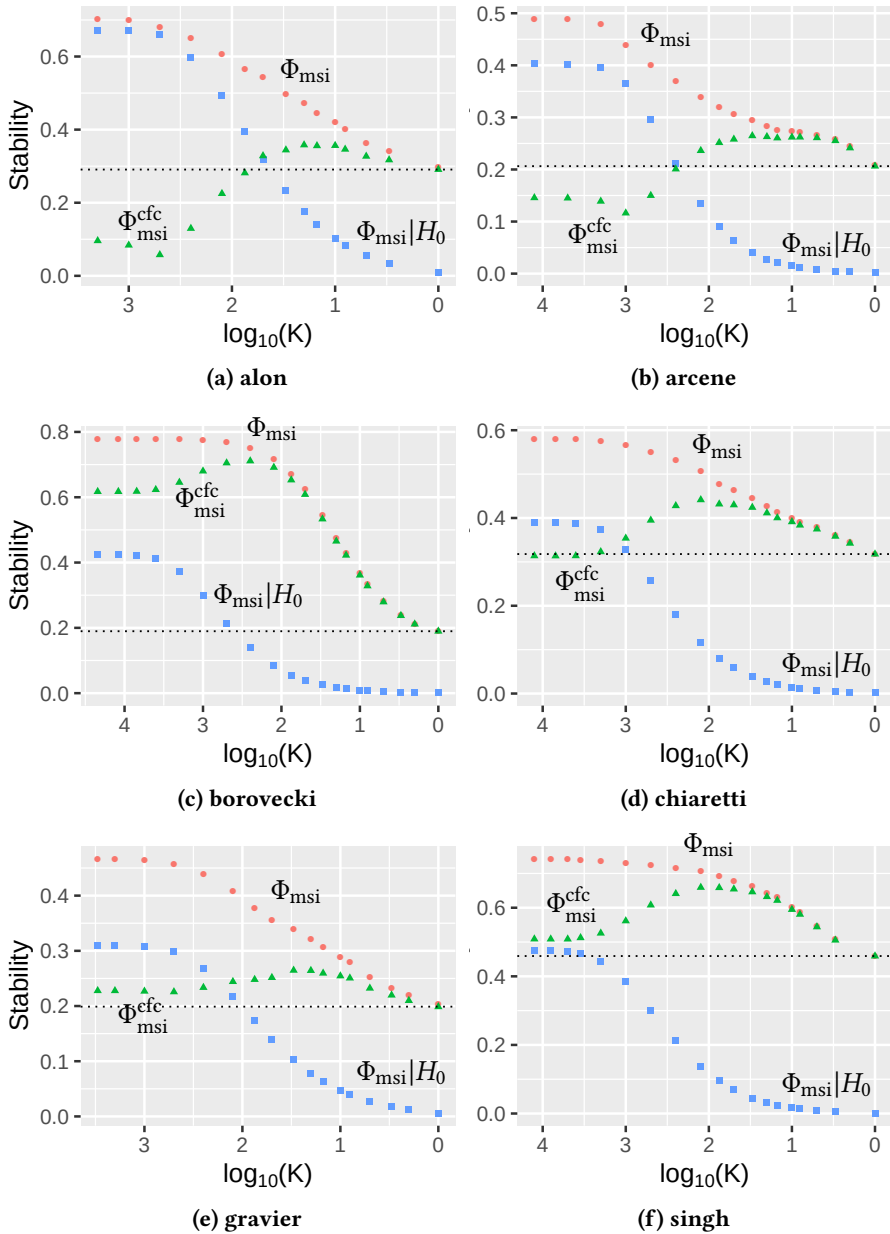


Figure D.5: Dependency of Φ_{msi} , $\Phi_{\text{msi}}|H_0$ and $\Phi_{\text{msi}}^{\text{cfc}}$ on the sparsification level K . The dotted line corresponds to the value of Φ_{iw} . When $K = 1$ ($\log_{10}(K) = 0$), feature correlations are ignored altogether and $\Phi_{\text{msi}} \approx \Phi_{\text{msi}}^{\text{cfc}} = \Phi_{\text{iw}}$. When the highest correlations for each feature are added to S according to Equation D.3, $\Phi_{\text{msi}}^{\text{cfc}}$ first increases as the selection algorithm tends to alternate between the most correlated features. Increasing K further results in $\Phi_{\text{msi}}|H_0$ quickly catching Φ_{msi} , which reduces $\Phi_{\text{msi}}^{\text{cfc}}$, sometimes even below Φ_{iw} .

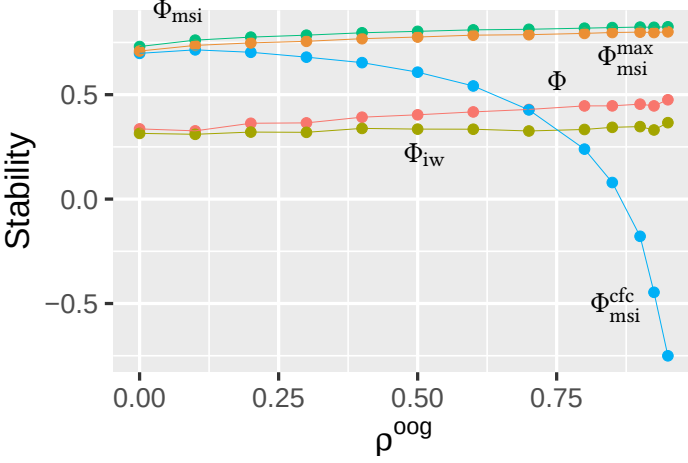


Figure D.6: Stability evolution in the simulated scenario studied in Figure D.1. The stability $\Phi_{\text{msi}}^{\text{max}}$ does not collapse when ρ^{oog} increases despite being (asymptotically) corrected for chance.

In conclusion, Φ_{msi} is not totally satisfactory, mostly when S is dense, as it lacks the correction for chance property. A high value of Φ_{msi} is not sufficient to conclude that the selection algorithm is stable. The corrected for chance measure $\Phi_{\text{msi}}^{\text{cfc}}$ is not fully satisfactory either, as it lacks the lower-bound property and sometimes produces dubious results. Sparsifying S to a certain level greatly mitigates this issue. However, this poses the question of choosing a specific value for the sparsity threshold. To combine the advantages of Φ_{msi} and $\Phi_{\text{msi}}^{\text{cfc}}$ while removing this threshold dependency, one can define stability as the maximum value of $\Phi_{\text{msi}}^{\text{cfc}}$ over all sparsity thresholds K . This comes at the additional computational cost of finding the optimal value for K . Nonetheless, it turns out that the heuristic $K = 0.2 \frac{d}{k}$ falls very close to the optimal value, at least on the 6 data sets considered here (where $d \geq 2000$ and with $k = 20$). Using this heuristic here works well without the additional computational cost. However, this need not be the case for all data sets, especially those with fewer input features.

Bibliography

- [Abe+10] T. Abeel, T. Helleputte, Y. Van de Peer, P. Dupont, and Y. Saeys. “Robust biomarker identification for cancer diagnosis with ensemble feature selection methods”. In: *Bioinformatics* 26.3 (2010), pp. 392–398.
- [Ale13] S. Alelyani. “On feature selection stability: A data perspective”. PhD thesis. Arizona State University, 2013.
- [Awa+12] W. Awada, T. M. Khoshgoftaar, D. Dittman, R. Wald, and A. Napolitano. “A review of the stability of feature selection techniques for bioinformatics data”. In: *International Conference on Information Reuse & Integration (IRI)*. IEEE. 2012, pp. 356–363.
- [Bel61] R. Bellman. “Adaptive Control Processes; A Guided Tour, Princeton Univ”. In: *Press, Nj* (1961).
- [BL97] A. L. Blum and P. Langley. “Selection of relevant features and examples in machine learning”. In: *Artificial intelligence* 97.1-2 (1997), pp. 245–271.
- [Bre+84] L. Breiman, J. Friedman, C. Stone, and R. Olshen. *Classification and Regression Trees*. Taylor & Francis, 1984. ISBN: 9780412048418.
- [Bre01a] L. Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.
- [Bre01b] L. Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.
- [Bre96] L. Breiman. “Bagging predictors”. In: *Machine learning* 24.2 (1996), pp. 123–140.
- [Bro+12] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján. “Conditional likelihood maximisation: a unifying framework for information theoretic feature selection”. In: *The journal of machine learning research* 13 (2012), pp. 27–66.
- [BS09] A.-L. Boulesteix and M. Slawski. “Stability and aggregation of ranked gene lists”. In: *Briefings in Bioinformatics* 10.5 (2009), pp. 556–568.

- [CV95] C. Cortes and V. Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995), pp. 273–297.
- [Dav+06] C. A. Davis, F. Gerick, V. Hintermair, C. C. Friedel, K. Fundel, R. Küffner, and R. Zimmer. "Reliable gene signatures for microarray classification: assessment of stability and performance". In: *Bioinformatics* 22.19 (2006), pp. 2356–2363.
- [DCA02] K. Dunne, P. Cunningham, and F. Azuaje. "Solutions to instability problems with sequential wrapper-based approaches to feature selection". In: *Journal of Machine Learning Research* (2002), pp. 1–22.
- [DD06] R. Díaz-Uriarte and S. A. De Andres. "Gene selection and classification of microarray data using random forest". In: *BMC bioinformatics* 7.1 (2006), pp. 1–13.
- [Dem06] J. Demšar. "Statistical comparisons of classifiers over multiple data sets". In: *The Journal of Machine learning research* 7 (2006), pp. 1–30.
- [DIB97] J. L. DeRisi, V. R. Iyer, and P. O. Brown. "Exploring the metabolic and genetic control of gene expression on a genomic scale". In: *Science* 278.5338 (1997), pp. 680–686.
- [Die98] T. G. Dietterich. "Approximate statistical tests for comparing supervised classification learning algorithms". In: *Neural computation* 10.7 (1998), pp. 1895–1923.
- [DKB03] B. Draper, C. Kaito, and J. Bins. "Iterative relief". In: *2003 Conference on Computer Vision and Pattern Recognition Workshop*. Vol. 6. IEEE. 2003, pp. 62–62.
- [DP05] C. Ding and H. Peng. "Minimum redundancy feature selection from microarray gene expression data". In: *Journal of Bioinformatics and Computational Biology* 3.02 (2005), pp. 185–205.
- [Efr+04] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. "Least angle regression". In: *The Annals of statistics* 32.2 (2004), pp. 407–499.
- [EP04] T. Evgeniou and M. Pontil. "Regularized multi-task learning". In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2004, pp. 109–117.
- [FHT10] J. Friedman, T. Hastie, and R. Tibshirani. "Regularization paths for generalized linear models via coordinate descent". In: *Journal of statistical software* 33.1 (2010), p. 1.
- [Fri37] M. Friedman. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance". In: *Journal of the american statistical association* 32.200 (1937), pp. 675–701.

- [GA11] R. Guzmán-Martínez and R. Alaiz-Rodríguez. “Feature selection stability assessment based on the jensen-shannon divergence”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2011, pp. 597–612.
- [GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- [GEW06] P. Geurts, D. Ernst, and L. Wehenkel. “Extremely randomized trees”. In: *Machine learning* 63.1 (2006), pp. 3–42.
- [GLM07] A. Genkin, D. D. Lewis, and D. Madigan. “Large-scale Bayesian logistic regression for text categorization”. In: *technometrics* 49.3 (2007), pp. 291–304.
- [Gol+99] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al. “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring”. In: *science* 286.5439 (1999), pp. 531–537.
- [Guy+02] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. “Gene selection for cancer classification using support vector machines”. In: *Machine Learning* 46.1-3 (2002), pp. 389–422.
- [Guy+08] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature extraction: foundations and applications*. Vol. 207. Springer, 2008.
- [GW16] W. W. B. Goh and L. Wong. “Evaluating feature-selection stability in next-generation proteomics”. In: *Journal of bioinformatics and computational biology* 14.05 (2016), p. 1650029.
- [Hai+13] B. Haibe-Kains, N. El-Hachem, N. J. Birkbak, A. C. Jin, A. H. Beck, H. J. Aerts, and J. Quackenbush. “Inconsistency in large pharmacogenomic studies”. In: *Nature* 504.7480 (2013), pp. 389–393.
- [Ham21] V. Hamer. *stabilitymaps: A domain analysis tool for feature selection*. R package. 2021.
- [HD19] V. Hamer and P. Dupont. “Explicit Control of Feature Relevance and Selection Stability Through Pareto Optimality”. In: *Proceedings of the 3rd International Tutorial & Workshop on Interactive Adaptive Learning (IAL2019) in conjunction with ECML-PKDD 2019*. Vol. 1. 2019, pp. 64–79.
- [HD20] V. Hamer and P. Dupont. “Joint optimization of predictive performance and selection stability.” In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. 2020, pp. 381–386.

- [HD21a] V. Hamer and P. Dupont. “An Importance Weighted Feature Selection Stability Measure”. In: *Journal of Machine Learning Research* 22.116 (2021), pp. 1–57.
- [HD21b] V. Hamer and P. Dupont. “Robust Selection Stability Estimation in Correlated Spaces”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021, pp. 446–461.
- [HT56] I. Heller and C. Tompkins. “An extension of a theorem of Dantzig’s”. In: *Linear inequalities and related systems* 38 (1956), pp. 247–254.
- [HY12] Y. Han and L. Yu. “A variance reduction framework for stable feature selection”. In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5.5 (2012), pp. 428–445.
- [Jav] JavaTpoint. *Support Vector Machine Algorithm*. URL: <https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm> (visited on 06/01/2022).
- [Jia+20] S. Jiang, Z. Song, O. Weinstein, and H. Zhang. *Faster dynamic matrix inverse for faster lps*. In *arXiv preprint*. 2020.
- [JOV09] L. Jacob, G. Obozinski, and J.-P. Vert. “Group lasso with overlap and graph lasso”. In: *Proceedings of the 26th annual international conference on machine learning*. 2009, pp. 433–440.
- [Jur+08] G. Jurman, S. Merler, A. Barla, S. Paoli, A. Galea, and C. Furlanello. “Algebraic stability indicators for ranked lists in molecular profiling”. In: *Bioinformatics* 24.2 (2008), pp. 258–264.
- [KKH07] P. Křížek, J. Kittler, and V. Hlaváč. “Improving stability of feature selection methods”. In: *International Conference on Computer Analysis of Images and Patterns*. Springer. 2007, pp. 929–936.
- [Kon94] I. Kononenko. “Estimating attributes: Analysis and extensions of RELIEF”. In: *European conference on machine learning*. Springer. 1994, pp. 171–182.
- [KPH05] A. Kalousis, J. Prados, and M. Hilario. “Stability of feature selection algorithms”. In: *International Conference on Data Mining (ICDM)*. IEEE. 2005, 8–pp.
- [KPH07] A. Kalousis, J. Prados, and M. Hilario. “Stability of feature selection algorithms: a study on high-dimensional spaces”. In: *Knowledge and Information Systems* 12.1 (2007), pp. 95–116.
- [KR+92] K. Kira, L. A. Rendell, et al. “The feature selection problem: Traditional methods and a new algorithm”. In: *Aaai*. Vol. 2. 1992, pp. 129–134.

- [Kun07] L. I. Kuncheva. “A stability index for feature selection.” In: *Artificial Intelligence and Applications*. 2007, pp. 421–427.
- [KV10] R. Kumar and S. Vassilvitskii. “Generalized distances between rankings”. In: *International Conference on World Wide Web (WWW)*. 2010, pp. 571–580.
- [KX10] S. Kim and E. P. Xing. “Tree-guided group lasso for multi-task regression with structured sparsity”. In: *ICML*. 2010.
- [Lau+13] L. Lausser, C. Müssel, M. Maucher, and H. A. Kestler. “Measuring and visualizing the stability of biomarker selection techniques”. In: *Computational Statistics* 28.1 (2013), pp. 51–65.
- [LCW16] Y. Li, C.-Y. Chen, and W. W. Wasserman. “Deep feature selection: theory and application to identify enhancers and promoters”. In: *Journal of Computational Biology* 23.5 (2016), pp. 322–336.
- [Lee+13] H. W. Lee, C. Lawton, Y. J. Na, and S. Yoon. “Robustness of chemometrics-based feature selection methods in early cancer detection and biomarker discovery”. In: *Statistical applications in genetics and molecular biology* 12.2 (2013), pp. 207–223.
- [LGV09] J. L. Lustgarten, V. Gopalakrishnan, and S. Visweswaran. “Measuring stability of feature selection in biomedical datasets”. In: *AMIA annual symposium proceedings*. Vol. 2009. American Medical Informatics Association. 2009, p. 406.
- [LM98] H. Liu and H. Motoda. *Feature extraction, construction and selection: A data mining perspective*. Vol. 453. Springer Science & Business Media, 1998.
- [Lok99] J. Lokhorst. “The lasso and generalised linear models”. In: *Honors Project, The University of Adelaide, Australia* (1999).
- [LPK21] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. “Explainable ai: A review of machine learning interpretability methods”. In: *Entropy* 23.1 (2021), p. 18.
- [Mar+15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin

- Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015.
- [MB10] N. Meinshausen and P. Bühlmann. “Stability selection”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.4 (2010), pp. 417–473.
- [Men02] S. Menard. *Applied logistic regression analysis*. 106. Sage, 2002.
- [MVB08] L. Meier, S. Van De Geer, and P. Bühlmann. “The group lasso for logistic regression”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 70.1 (2008), pp. 53–71.
- [NB16] S. Nogueira and G. Brown. “Measuring the stability of feature selection”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. Springer. 2016, pp. 442–457.
- [NB99] C. Nadeau and Y. Bengio. “Inference for the generalization error”. In: *Advances in neural information processing systems* 12 (1999).
- [Nem63] P. B. Nemenyi. *Distribution-free multiple comparisons*. Princeton University, 1963.
- [NSB17a] S. Nogueira, K. Sechidis, and G. Brown. “On the stability of feature selection algorithms”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 6345–6398.
- [NSB17b] S. Nogueira, K. Sechidis, and G. Brown. “On the use of Spearman’s rho to measure the stability of feature rankings”. In: *Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*. Springer. 2017, pp. 381–391.
- [PLD05] H. Peng, F. Long, and C. Ding. “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy”. In: *IEEE Transactions on pattern analysis and machine intelligence* 27.8 (2005), pp. 1226–1238.
- [PVD+13] J. Paul, M. Verleysen, P. Dupont, et al. “Identification of statistically significant features from random forests”. In: *ECML workshop on Solving Complex Machine Learning Problems with Ensemble Methods*. 2013, pp. 69–80.
- [PVD12] J. Paul, M. Verleysen, and P. Dupont. “The stability of feature selection and class prediction from ensemble tree classifiers.” In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*. 2012.

- [Qiu+06] X. Qiu, Y. Xiao, A. Gordon, and A. Yakovlev. “Assessing stability of gene selection in microarray data analysis”. In: *BMC bioinformatics* 7.1 (2006), pp. 1–13.
- [RK03] M. Robnik-Šikonja and I. Kononenko. “Theoretical and empirical analysis of ReliefF and RReliefF”. In: *Machine learning* 53.1 (2003), pp. 23–69.
- [RMM15] D. Roy, K. S. R. Murty, and C. K. Mohan. “Feature selection using deep neural networks”. In: *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2015, pp. 1–6.
- [Rot04] V. Roth. “The generalized LASSO”. In: *IEEE transactions on neural networks* 15.1 (2004), pp. 16–28.
- [RVB15] N. Riquelme, C. Von Lüken, and B. Baran. “Performance metrics in multi-objective optimization”. In: *2015 Latin American computing conference (CLEI)*. IEEE. 2015, pp. 1–11.
- [SAV08a] Y. Saeys, T. Abeel, and Y. Van de Peer. “Robust feature selection using ensemble feature selection techniques”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. Springer. 2008, pp. 313–325.
- [SAV08b] Y. Saeys, T. Abeel, and Y. Van de Peer. “Robust feature selection using ensemble feature selection techniques”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. Springer. 2008, pp. 313–325.
- [Sch+95] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. “Quantitative monitoring of gene expression patterns with a complementary DNA microarray”. In: *Science* 270.5235 (1995), pp. 467–470.
- [SDC03] M. R. Segal, K. D. Dahlquist, and B. R. Conklin. “Regression approaches for microarray data analysis”. In: *Journal of Computational Biology* 10.6 (2003), pp. 961–980.
- [Sec+19] K. Sechidis, K. Papangelou, S. Nogueira, J. Weatherall, and G. Brown. “On The Stability of Feature Selection in the Presence of Feature Correlations”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*. 2019.
- [Sha+12] A. A. Shanab, T. M. Khoshgoftaar, R. Wald, and A. Napolitano. “Impact of noise and data sampling on stability of feature ranking techniques for biological datasets”. In: *International Conference on Information Reuse and Integration (IRI)*. IEEE. 2012, pp. 415–422.

- [Shi+06] L. Shi, L. H. Reid, W. D. Jones, R. Shippy, J. A. Warrington, S. C. Baker, P. J. Collins, F. De Longueville, E. S. Kawasaki, K. Y. Lee, et al. “The MicroArray Quality Control (MAQC) project shows inter-and intraplatform reproducibility of gene expression measurements”. In: *Nature biotechnology* 24.9 (2006), p. 1151.
- [SIL07] Y. Saeys, I. Inza, and P. Larrañaga. “A review of feature selection techniques in bioinformatics”. In: *Bioinformatics* 23.19 (2007), pp. 2507–2517.
- [SK03] S. K. Shevade and S. S. Keerthi. “A simple and efficient algorithm for gene selection using sparse logistic regression”. In: *Bioinformatics* 19.17 (2003), pp. 2246–2253.
- [SK10] G. Stiglic and P. Kokol. “Stability of ranked gene lists in large microarray analysis studies”. In: *Journal of biomedicine and biotechnology* 2010 (2010).
- [SMS03] R. L. Stears, T. Martinsky, and M. Schena. “Trends in microarray analysis”. In: *Nature medicine* 9.1 (2003), pp. 140–145.
- [SN10a] P. Somol and J. Novovicova. “Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.11 (2010), pp. 1921–1939.
- [SN10b] P. Somol and J. Novovičová. “Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.11 (2010), pp. 1921–1939.
- [TAL14] J. Tang, S. Alelyani, and H. Liu. “Feature selection for classification: A review”. In: *Data Classification: Algorithms and Applications* (2014), p. 37.
- [Tib96] R. Tibshirani. “Regression shrinkage and selection via the lasso”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), pp. 267–288.
- [UPC20] A. Urkullu, A. Pérez, and B. Calvo. “Statistical model for reproducibility in ranking-based feature selection”. In: *Knowledge and Information Systems* (2020), pp. 1–32.
- [Urb+18] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore. “Relief-based feature selection: Introduction and review”. In: *Journal of biomedical informatics* 85 (2018), pp. 189–203.
- [VR12] J. Vogt and V. Roth. “A complete analysis of the l₁, p group-lasso”. In: *arXiv preprint arXiv:1206.4632* (2012).

- [Wan+07] J. Z. Wang, Z. Du, R. Payattakool, P. S. Yu, and C.-F. Chen. “A new method to measure the semantic similarity of GO terms”. In: *Bioinformatics* 23.10 (2007), pp. 1274–1281.
- [Wel47] B. L. Welch. “The generalization of student’s’ problem when several different population variances are involved”. In: *Biometrika* 34.1/2 (1947), pp. 28–35.
- [WKS12] R. Wald, T. M. Khoshgoftaar, and A. A. Shanab. “The effect of measurement approach and noise level on gene selection stability”. In: *2012 IEEE International Conference on Bioinformatics and Biomedicine*. IEEE. 2012, pp. 1–5.
- [YDL08] L. Yu, C. Ding, and S. Loscalzo. “Stable feature selection via dense feature groups”. In: *International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)*. 2008, pp. 803–811.
- [YHB11] L. Yu, Y. Han, and M. E. Berens. “Stable gene selection from microarray data via sample weighting”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9.1 (2011), pp. 262–272.
- [YL06] M. Yuan and Y. Lin. “Model selection and estimation in regression with grouped variables”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1 (2006), pp. 49–67.
- [Yu+10] G. Yu, F. Li, Y. Qin, X. Bo, Y. Wu, and S. Wang. “GOSemSim: an R package for measuring semantic similarity among GO terms and gene products”. In: *Bioinformatics* 26.7 (2010), pp. 976–978.
- [ZBT07] E. Zitzler, D. Brockhoff, and L. Thiele. “The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration”. In: *International Conference on Evolutionary Multi-Criterion Optimization*. Springer. 2007, pp. 862–876.
- [ZH05] H. Zou and T. Hastie. “Regularization and variable selection via the elastic net”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320.
- [Zha+09] M. Zhang, L. Zhang, J. Zou, C. Yao, H. Xiao, Q. Liu, J. Wang, D. Wang, C. Wang, and Z. Guo. “Evaluating reproducibility of differential expression discoveries in microarray studies by considering correlated molecular changes”. In: *Bioinformatics* 25.13 (2009), pp. 1662–1668.
- [ZRS08] M. Zucknick, S. Richardson, and E. A. Stronach. “Comparing the characteristics of gene expression profiles derived by univariate and multivariate classification methods”. In: *Statistical applications in genetics and molecular biology* 7.1 (2008).

- [ZRY09] P. Zhao, G. Rocha, and B. Yu. “The composite absolute penalties family for grouped and hierarchical variable selection”. In: *The Annals of Statistics* 37.6A (2009), pp. 3468–3497.
- [ZT99] E. Zitzler and L. Thiele. “Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach”. In: *IEEE transactions on Evolutionary Computation* 3.4 (1999), pp. 257–271.
- [ZY17] Y. Zhang and Q. Yang. “A survey on multi-task learning”. In: *arXiv preprint arXiv:1707.08114* (2017).