جامعة سيدي محمد بن عبد الله بفاس
UNIVERSITE SIDI MOHAMED BEN ABDELLAH DE FES
كلية العلوم ظهر المهراز فاس
FACULTE DES SCIENCES DHAR EL MAHRAZ FES

N° d'Ordre : 72**/2021**

Formation Doctorale : Sciences et Technologies de l'Information et de Communication (STIC)

Discipline : Machine learning and data mining

Spécialité : Informatique

Laboratoire : Laboratoire de Physique Appliquée, Informatique et Statistiques (LPAIS)

THESE DE DOCTORAT

Présentée par

Yassine Akhiat

Feature Selection Methods for High Dimensional Data

Soutenue le 31 / 12 / 2021 devant le jury composé de :

Pr. El Habib NFAOUI	***Faculté des Sciences Dhar El Mahraz- Fès***	***Président***
Pr. Brahim OUHBI	***Ecole Nationale Supérieure des Arts et Métiers Meknès***	***Rapporteur***
Pr. Ismail BERRADA	***Université Mohammed VI Polytechnique- Benguerir***	***Rapporteur***
Pr. Mohamed EL FAR	***Faculté des Sciences Dhar El Mahraz- Fès***	***Rapporteur***
Pr. Bouchra FRIKH	***Ecole Supérieure de Technologie- Fès***	***Examinateur***
Pr. Anass BOUAYAD	***Faculté des Sciences Dhar El Mahraz- Fès***	**Examinateur**
Pr. Ahmed ZINEDINE	***Faculté des Sciences Dhar El Mahraz- Fès***	**Directeur de thèse**
Pr. Mohamed CHAHHOU	***Faculté des Sciences, UAE-Tetouan***	**Co-directeur de thèse**

Année universitaire : 2021-2022

SIDI MOHAMED BEN ABDELAH UNIVERSITY

PHD THESIS

Feature selection methods for high dimensional data

Author :
Yassine AKHIAT

Supervisor :
Prof. Ahmed ZINEDINE
Co-supervisor :
Prof. Mohamed CHAHHOU

A thesis submitted in fulfillment of the requirements for the degree of Philosophy Doctor (PhD)

in the

"Computer Science and Informatics" specialty

This thesis work is dedicated to my parents For their :
endless love,
support and encouragement…

Acknowledgements

"We all change. When you think about it, we're all different people all through our lives, and that's okay, that's good, you gotta keep moving, so long as you remember all the people that you used to be. I will not forget one line of this. Not one day. I swear. I will always remember when the Doctor was me." The Doctor (Doctor Who, The Time of the Doctor)

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study. For this reason, I will take this opportunity to express my immense gratitude to each one of them.

First and foremost, I would to like express my deepest gratitude to my professor, **Pr. Ahmed Zinedine** for his guidance during my thesis. His perpetual energy and enthusiasm motivated me to put in my best effort. In addition, He was always accessible and willing to help me with all the queries and difficulties I faced. I would also like to thank him for his personal and human qualities which have also contributed a lot to the accomplishment of this work.

Special thanks to my Co-supervisor, **Pr. Mohamed Chahhou**. It would not have been possible to write this thesis if he hadn't kindled an interest in me to start my thesis. My utmost gratitude goes to him whose sincerity and encouragement has been my driving force.

Particular thanks are given to all my friends who were willing to put into time and effort to proof read various drafts of the technical writings related to this thesis especially **Mr. Hafid Tahboun**.

I would to extend my warm thanks to my family, friends and colleagues who gave me continuous encouragement and uninterrupted stimulus throughout all my years at university. I especially appreciate the unconditional support of my parents without whom I would not have made it through my PhD studies. No words would be enough to reward your sacrifices and patience.

Thank you all.

Abstract

With the advent of high-dimensional data, typically many features are irrelevant, redundant and noisy for a given learning task, having harmful consequences in terms of performance and/or computational cost. Moreover, a large number of features requires a large amount of memory or storage space. Therefore, reducing the dimensionality of data has become indispensable in real world scenarios to successfully build simpler and accurate models, improving data-mining performance and enhancing the interpretability of models. Feature selection is one of the most fundamental dimensionality reduction techniques in many areas including: text classification, image recognition, microarrays and bioinformatics. It can be defined as the process of identifying and selecting the relevant features and removing the irrelevant and noisy ones, with the goal of obtaining a small subset (lower dimensionality) of features that describes properly a given problem maintaining or even improving the performance.

This thesis work fall within the framework of dimensionality reduction especially, it is devoted to feature selection research and its application to real high dimensional data where several specific challenges are addressed. The first part of this work presents an in-depth analysis of feature selection procedure and provides a critical review of its state-of-the-art methods and categorization so as to supplement insights and recommendations to help researchers and enlighten readers. Moreover, we have provided some guidelines for efficient performing FS. We also apply some well-know feature selection methods on several benchmarking datasets to demonstrate the applicability of feature selection techniques and to expose their merits and demerits. After performing an in-depth analysis of existing feature selection methods, the second part of this thesis focused on proposing novel feature selection techniques aiming to solve some of the problems detected in the field.

The first proposal is a hybrid filter-wrapper feature selection method based on pairwise features evaluation. It benefits from the filters' simplicity as well as the high performance of wrappers. The second proposal is relying on graph representation where each node corresponds to each feature, and the pairwise score between two features is used as the weight of the edge between two nodes. Since the stability of feature selection algorithm is an overlooked problem, the third proposed algorithm consists of using ensemble technique to ensure the stability of feature selection. Sometimes random forest (RF) model over-fits on noisy features which lead to choosing the noisy features as the informative variables and eliminating the significant ones. To address this problem, we have proposed a new variant of RF that provides unbiased variable selection where a noisy feature trick is used. Then, the best subset of features is selected out of the best-ranked feature regarding the Gini impurity of this new variant of RF.

Finally, in the fifth contribution, we have developed a new reinforcement learning-based approach for feature selection. This latter acts like an agent that traverses the feature space to continuously learn and explore rules (sub-sets) to properly select the best performing features.

Our proposals are evaluated and assessed on a common experimental design that considers large standard and well-known datasets for feature selection which are publicly available on UCI repository and kaggle platform. The experimental results support the validity of our contributions.

Keywords: Machine learning, Data Mining, Dimension Reduction, Feature Selection, High Dimensional Data

Résumé

Avec la croissance rapide des données de grande dimension, de nombreuses attributs sont généralement non pertinentes, redondantes et bruyantes pour une tâche d'apprentissage donnée, ce qui a des conséquences néfastes en termes de performances et / ou de coût de calcul. De plus, un grand nombre d'attributs nécessite une grande quantité de mémoire ou d'espace de stockage. Par conséquent, la réduction de la dimensionnalité des données est devenue indispensable dans les scénarios du monde réel pour réussir à construire des modèles plus simples et précis, à améliorer les performances d'exploration de données et à améliorer l'interprétabilité des modèles. La sélection de caractéristiques est l'une des techniques de réduction de dimensionnalité les plus fondamentales dans de nombreux domaines, notamment : la classification de texte, la reconnaissance d'images, les puces à ADN et la bio-informatique. Il peut être défini comme le processus d'identification et de sélection des caractéristiques pertinentes et de suppression de celles qui ne sont pas pertinentes et bruyantes, dans le but d'obtenir un petit sous-ensemble (dimensionnalité inférieure) de caractéristiques qui décrit correctement un problème donné en maintenant ou même en améliorant les performances.

Ce travail de thèse s'inscrit dans le cadre de la réduction de la dimensionnalité en particulier, il est consacré à la recherche de sélection des attributs fonctionnels et à son application à des données réelles de haute dimension où plusieurs défis spécifiques sont abordés. La première partie de ce travail présente une analyse approfondie de la procédure de sélection des fonctionnalités et fournit un examen critique de ses méthodes de pointe et de sa catégorisation afin de fournir aux lecteurs et aux chercheurs une vision claire, des recommandations pertinente et quelques lignes directrices pour une sélection d'attributs efficace. La deuxième partie de cette thèse est consacrée à la proposition de nouvelles techniques de sélection de fonctionnalités visant à résoudre certains des problèmes détectés sur le terrain. Dans ce stade, nous avons proposé cinq contributions différentes pour améliorer la sélection des attributs dans les grandes dimensions. Les expérimentations menées prouvent que les résultats sont prometteurs.

La première proposition est une méthode de sélection de caractéristiques d'enveloppe de filtre hybride basée sur l'évaluation de caractéristiques par paires. Il bénéficie de la simplicité des filtres ainsi que de la haute performance des méthodes wrappers. La deuxième proposition repose sur une représentation graphique où chaque nœud correspond à chaque caractéristique, et le score par paire entre deux caractéristiques est utilisé comme poids de l'arête entre deux nœuds. Puisque la stabilité de l'algorithme de sélection de caractéristiques est un problème souvent négligé dans la littérature, le troisième algorithme proposé consiste à utiliser une technique d'ensemble pour assurer la stabilité de la sélection de caractéristiques.

Parfois, le modèle de forêt aléatoire (RF) surajuste les caractéristiques bruyantes, ce qui conduit à choisir les caractéristiques bruyantes comme variables informatives et à éliminer les plus significatives. Pour résoudre ce problème, nous avons proposé une nouvelle variante de RF qui fournit une sélection de variables non biaisée où une astuce de fonctionnalité bruyante est utilisée. Ensuite, le meilleur sous-ensemble de caractéristiques est sélectionné parmi les caractéristiques les mieux classées en ce qui concerne l'impureté Gini de cette nouvelle variante de RF. Enfin, dans la cinquième contribution, nous avons développé une nouvelle approche basée sur l'apprentissage par renforcement pour la sélection de caractéristiques. Ce dernier agit comme un agent qui parcourt l'espace des fonctionnalités pour apprendre et explorer en permanence des règles (sous-ensembles) afin de sélectionner correctement les fonctionnalités les plus performantes.

Nos propositions sont évaluées sur une configuration expérimentale de référence qui prend en compte de grands ensembles de données standard et largement utilisé pour la sélection des attributs qui sont accessibles au public sur le référentiel UCI et la plate-forme kaggle. Les résultats expérimentaux confirment la validité de nos contributions.

Mots clés : Apprentissage Automatique,Fouille de Données, Réduction de Dimension, Sélection des Attributs, Réduction de Dimension, Données de Grande Dimension

ملخص

في عصرنا هذا، عصر البيانات الضخمة، غالبا ما تكون العديد من الميزات والخصائص غير مهمة أو متكررة وغالبا ما تؤثر سلبا على خوارزميات الذكاء الاصطناعي والتعلم الآلي ولها عواقب ضارة من حيث الأداء و / أو التكلفة الحسابية. علاوة على ذلك، يتطلب العدد الكبير والمتزايد للميزات مساحة تخزين كبيرة جدا. لذلك، أصبح تقليل وتقليص أبعاد البيانات أمرًا لا غنى عنه في سيناريوهات العالم الحقيقي لبناء نماذج أبسط وأدق، وتحسين أداء استخراج البيانات وتعزيز إمكانية تفسير النماذج. يعد انتقاء الميزات أحد أهم تقنيات تقليل الأبعاد الأساسية في العديد من المجالات بما في ذلك: تصنيف النص والتعرف على الصور والمصفوفات الدقيقة والمعلوماتية الحيوية. يمكن تعريفها على أنها عملية تحديد وانتقاء الميزات الوجيهة وإزالة الميزات غير الوجيهة والمشوشة، بهدف الحصول على مجموعة فرعية صغيرة (أبعاد أقل) من الميزات التي تصف بشكل صحيح مشكلة معينة والحفاظ على الأداء أو حتى تحسينه.

تندرج هذه الرسالة ضمن إطار تقليل الأبعاد على وجه الخصوص، فهي مكرسة لبحث انتقاء الميزات وتطبيقه على بيانات حقيقية عالية الأبعاد حيث يتم تناول العديد من التحديات المحددة. يقدم الجزء الأول من هذا العمل تحليلًا معمقًا لإجراءات الانتقاء ويوفر مراجعة نقدية لأساليبها الحديثة وتصنيفها وذلك لتوضيح الرؤية وتقديم توصيات لمساعدة الباحثين وتنوير القراء. علاوة على ذلك، قدمنا بعض الإرشادات لتعزيز استقرار وكفاءة خوارزميات انتقاء الميزات. بالإضافة الى ذلك، تم تطبيق بعض طرق وتقنيات انتقاء الميزات على العديد من مجموعات البيانات المعيارية لإثبات قابلية تطبيقها ولإظهار مزاياها وعيوبها. بعد إجراء تحليل معمق لطرق انتقاء الميزات الحديثة، يركز الجزء الثاني من هذه الأطروحة على اقتراح خوارزميات جديدة لانتقاء الميزات الوجيهة والتي تهدف إلى حل بعض المشكلات المكتشفة في المجال.

الكلمات المفاتيح: انتقاء الميزات، الغابة العشوائية، الخوارزمية، تقليل الابعاد، التعلم الالي، الذكاء الاصطناعي.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	**A**rtificial **I**ntelligence
FS	**F**eature **S**election
EFS	**E**nsemble **F**eature **S**election
MI	**M**utual **I**nformation
ML	**M**achine **L**earning
DM	**D**ata **M**ining
FE	**F**eature **E**xtraction
DR	**R**imensionality **R**eduction
FwS	**F**orward **S**election
KDD	**K**nowledge **D**idcovery in **D**ata
BwS	**B**ackward **S**election
DT	**D**ecision **T**ree
DL	**D**eep **L**earning
PR	**P**attern **R**ecognition
RFE	**R**ecursiv **F**eature **E**limination
SVM	**S**upport **V**ector **M**achine
KNN	**K** **N**earest **N**eighbor
GFS	**G**raph**F**eature **S**election
NRF	**N**oisy **R**andom **F**orest
RF	**R**andom **F**orest
TSM	**T**ransition **S**imilarity **M**easure
ECMBF	**E**fficient **C**orrelation **M**easure **B**ased **F**ilter
MRmMC	**M**aximum **R**elevance **m**inimum **M**ulti **C**ollinearity
CFS	**C**orrelation based **F**eature **S**election
mRmR	**m**inimum **R**edundancy **m**aximum **R**elevance
FS-P	**P**airwise **F**eature **S**election
KDD	**K**nowledge **D**iscovery in **D**atabase
ROC	**R**eceiver **O**perating **C**haracteristic
AUC-ROC	**A**rea **U**nder the **ROC** **C**urve
RMSE	**R**oot **M**ean **S**quare **E**rror

General Introduction

“

“Predicting the future isn’t magic, it’s artificial intelligence.”

Dave Waters.

General Introduction

Context and Motivation

With the rapid growth of modern technologies in the last two decades, the world has become increasingly more instrumented, interconnected, and intelligent. The limitless number of computer and Internet applications has caused an exponential increase in the amount of generated data in a variety of domains. In domains such as social media, healthcare, marketing and bioinformatics, the data provided may not only be huge in terms of the data samples (instances or examples), but also in terms of feature (characteristics) dimensionality. Therefore, the use of data mining and machine learning tools becomes a mandatory for automatically extracting knowledge, insights and identifying hidden patterns from data (Knowledge discovery in database KDD).

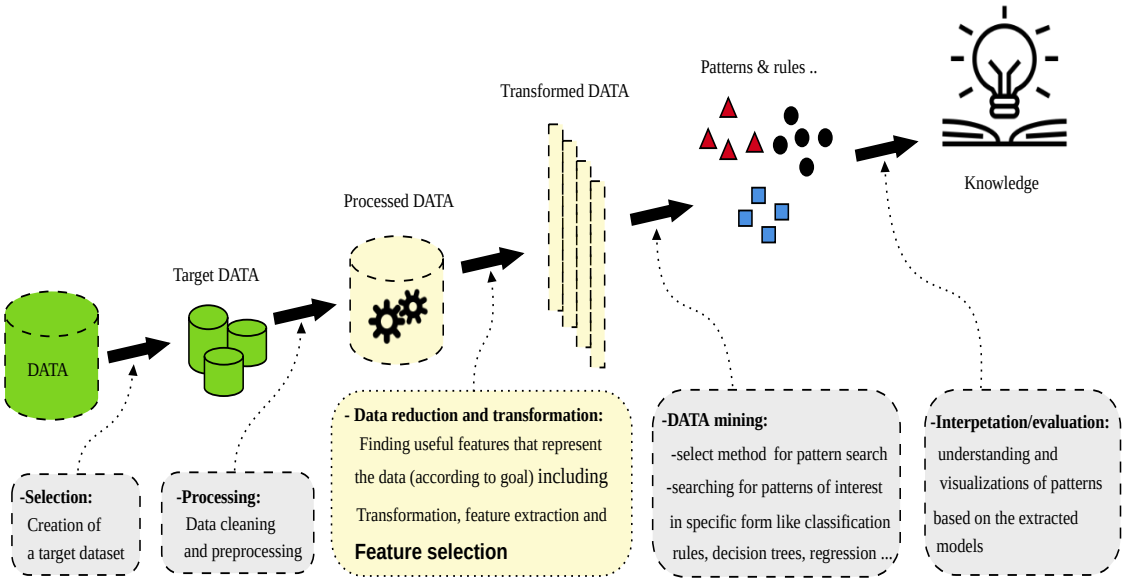


Figure 1 – The main steps of Knowledge discovery in database process.

Pattern recognition is the process of recognizing patterns using machine learning algorithms. It can be defined as the classification of data based on the knowledge extracted from patterns. Pattern recognition has a strong applied aspect in many domains such as text classification, face recognition, spam email identification, medicine, recognition tasks in biology, economics, astronomy, etc. A pattern is a vector of many observations (features) and characteristics of a different type. Since the volume and complexity of data are in a continuous growth, the massive amount of data has bombarded machine learning researchers with a plethora of

unprecedented challenges; making the learning task more complex and computationally more demanding. Applying data mining and machine learning algorithms in high-dimensional data, learning algorithms performance may degrade due to over-fitting problem Roelofs et al. (2019); Ying (2019). Given the existence of a large number of features, machine learning models become intricately complicated to interpret as their complexity increases leading to a less generalization ability.

Data mining can take advantage of dimensionality reduction tools, which is a major step of data pre-processing, to reduce the high dimensionality of data Li et al. (2017). Dimensionality reduction can be categorized into feature extraction and feature selection (see figure 2) Mitra et al. (2002); Liu et al. (2004). Feature extraction aims at transforming the original feature space to a new reduced one, where features lose their meaning due to the transformation Guyon et al. (2008a). In contrast to feature extraction, feature selection is the process of identifying the relevant features and removing the irrelevant and redundant ones, intending to obtain the best performing subset of original features without any transformation. Thus, the constructed learning models using the selected subset of features are more interpretable and readable Ghojogh et al. (2019). This gives preference to the reliable applicability of feature selection as an effective alternative prioritized over feature extraction in many real-world datasets. The main reasons for applying feature selection are basically the following:

- **The facilitation of models interpretation.**
  The interpretability has become an indispensible aspect of ML models in a plenty of domains where model performance is not enough to trust the model decisions. In the majority of real-world application, it is also necessary to know why certain decision or prediction was made especially in healthcare and financial services as well as in other strictly regulated domains to leverage ML systems for high-stakes decisions that deeply impact both human lives and society Rudin (2018), which pushes the demand for model interpretability even further. Feature selection can perfectly enhance model interpretability by reducing the feature space which may help us visualize, understand and see what might be affecting a model to take specific decisions.
- **The reduction of resources requirement (short training time, small storage capacity, etc.).**
  Feeding algorithms with smaller feature subsets generally leads to a fast execution and less storage capacity.
- **Avoiding the curse of dimensionality.**
  One of the fundamental motivations for feature selection is the curse of dimensionality especially when dealing with small sample size and large number of feature situations. Reducing feature space massively helps to provide better classification accuracy due to finite sample size effects Jain and Chandrasekaran (1982).
- **Avoiding over-fitting problem, automatically leading to a better model generalization.**

Reducing the amount of redundant/noisy data leads automatically to reduce the opportunity to make decisions based on noise. Thus, the overfitting problem is minimized.

- **Improving accuracy: less noise in data which means an improved modeling accuracy.**
  Too much data is not always good for machine learning algorithms. The presence of noisy and uninformative features can deteriorate the performance of learning algorithms. Therefore, eliminating the useless feature first, can improve accuracy.

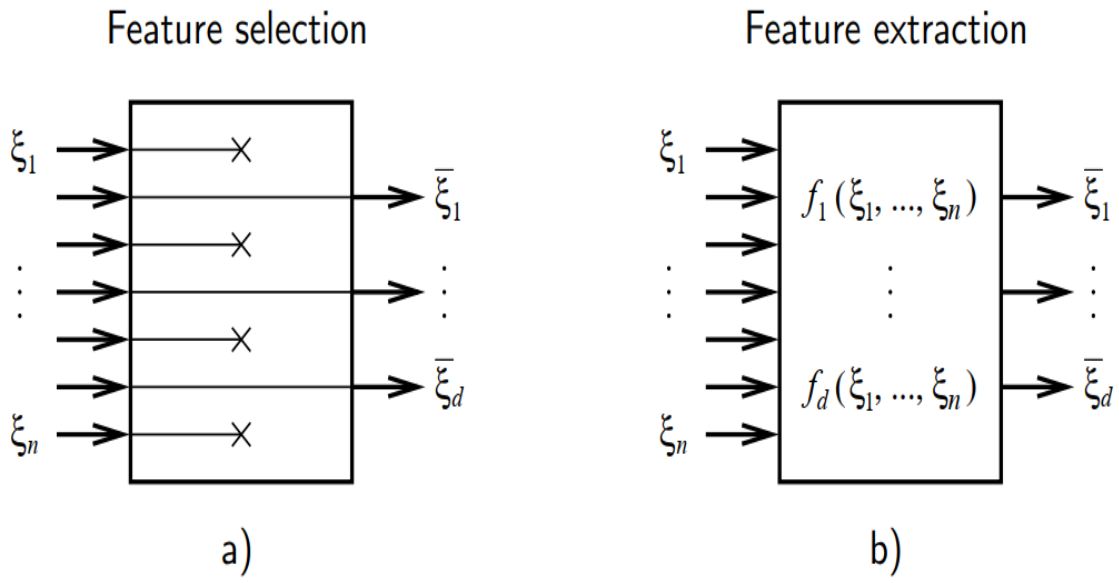


Figure 2 – Feature selection VS Feature extraction

Feature selection is an active research filed in machine learning as it is an important pre-processing, finding success in different real problem applications. In general, feature selection algorithms are categorized into supervised, Semi-supervised and Unsupervised feature selection according to the availability of label information. Supervised feature selection methods usually come in three flavors: Filter, Wrapper and Embedded approach.

Filter Methods relies on the relationship between features and the class label (such as distance, dependency, correlation, etc.) to assess the importance of features. This category is a pre-processing step, which is independent from the induction algorithm. Filters are known by their ease of use and low computational cost. On the contrary, Wrapper approach generates models with subsets of features. Then, it uses prediction performance as a criterion function to lead the search for the best feature subset. This approach takes into account the interactions between features. Generally, Wrappers achieve better performance than some Filter methods. Embedded approach performs feature selection by implication while simultaneously constructing models, which makes them less costly in terms of execution time than wrappers.

The wrapper approach has been mostly avoided in the literature Canedo and Marono (2014) due to its high computational cost. As the number of features increases, the feature subset space grows exponentially. The presence of ten thousands of features is a critical aspect. Besides, wrapper methods are more likely to over-fit, especially to small size datasets. As a result, the tendency is to focus on new alternatives such as hybrid or ensemble methods where hybrid approach tries to combine both filter and wrapper approaches while ensemble approach relies on ensemble techniques and its relation with feature selection. For this reason, the contributions of this thesis work fall into embedded, hybrid and ensemble category.

Our first proposal is related to hybrid approach. A filtering stage is introduced prior to feature selection. Then, a pairwise feature selection evaluation is performed to improve the performance of the classifier and to tackle the redundancy problem. Despite the fast execution and the ability of handling redundancy dilemma, the proposed pairwise method can not capture the third or higher order interactions that may exist between features Hindawi (2013). This problem is related to how to represent the feature space while preserving the maximum existing underlying interactions of higher order. This has motivated us to use Graph theory, which is an important tool for analyzing data with structure dependency. Two methods are developed in this thesis relying on graph structure. Each node of the graph corresponds to one feature, and each edge has a weight corresponding to the interaction information among features connected by that edge.

An overlooked problem is the stability of feature selection algorithms. Stability selection can be seen as the consistency of a given selector to yield a consistent feature subset when data instances are perturbed by adding/removing some training samples Haury et al. (2011); Dunne et al. (2002); Somol and Novovičová (2010); Yang and Mao (2010); Bolón-Canedo and Alonso-Betanzos (2019). If the algorithm produces a different subset for any perturbations in the training data, then the algorithm becomes unreliable for feature selection. For this reason, we adopted ensemble technique which is another interesting line of research in classification. Based on the proverb:"Two heads are better than one", which means that a set of experts is better than a single expert, we have propose two ensemble feature selection methods for the sake of enhancing the stability selection, therefore, producing accurate, reliable and more stable selectors.

In fact, the large number of features always hides complex and hard underlying interactions to reveal. Therefore, we have solved the feature selection problem using Reinforcement Learning paradigm as can be used to learn effective policies for complex tasks intending to identify the best performing subset. We have formulated the feature space (state space) as a Markov Decision Process (MDP) where the state space is controlled using a decision tree branches.

To better understand the structure and the main contributions of this thesis work, the following chart 3 summarizes and illustrates the serious problems and issues detected in the feature selection literature as well as the proposed contributions.

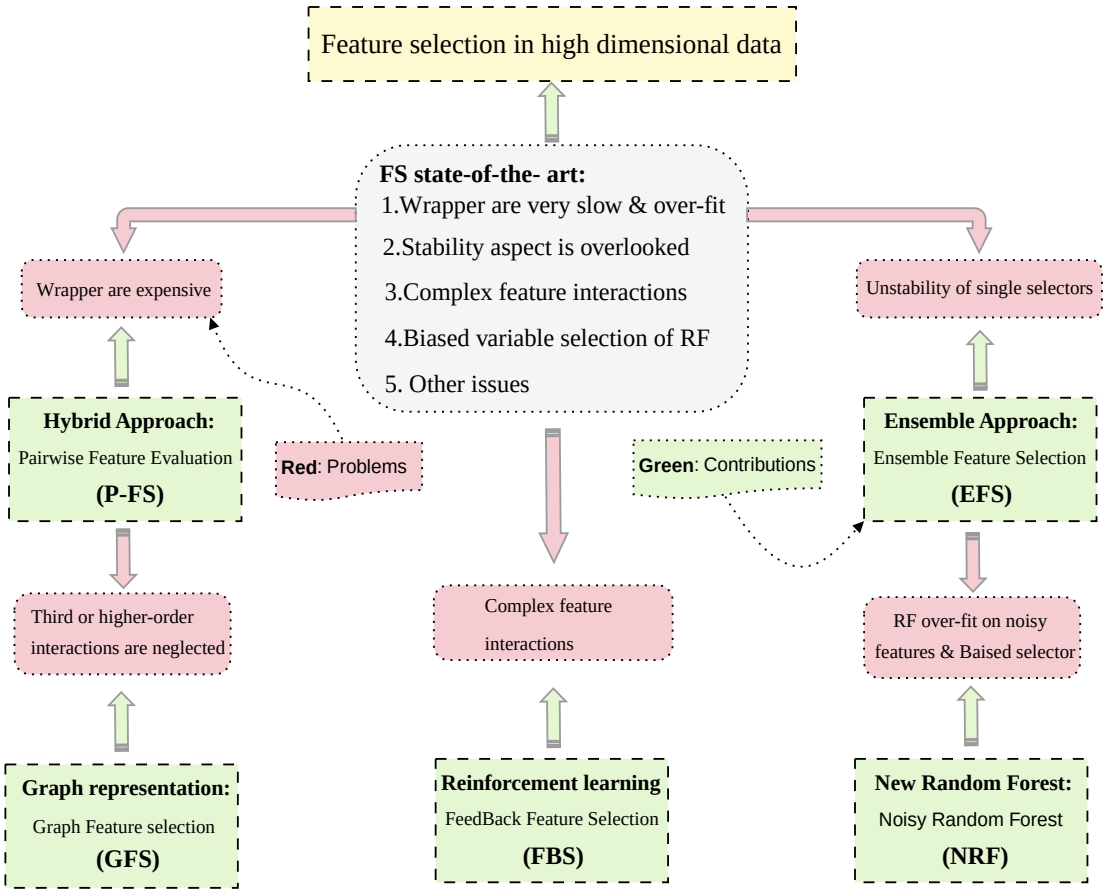


Figure 3 – An overview of the main feature selection issues VS the proposed contributions of the thesis.

Research Goals

The aim of this thesis work is to propose new methods in Feature Selection domains so as to find and identify optimal feature subset that accurately classifies and distinguishes between instances of different classes (enhance classification ability). This automatically enables learning algorithms to operate more effectively and rapidly. A typical Feature Selection system can be seen as the combination of many components, including subset generation, subset evaluation, stopping criteria and subset validation, which has to deal with the aforementioned specific challenges (redundancy, irrelevance, noise, high dimensional data, etc).

We consider that the improvement of such fundamental aspects of the usefulness of feature selection algorithm has to take into account the problems lying in each of the aforementioned components. It is a matter of proposing, designing, and evaluating a FS system which is able to automatically detect and select relevant variables only and discard irrelevant, noisy and redundant ones. To achieve this purpose, we have pursued the following research goals:

- **Research Goal 1**: is the proposal of hybrid feature selection methods for supervised classification tasks. The suggested system makes use of both the

rapidity and the exploitation of filters' simplicity as well as the qualitative performance of wrappers.

- **Research Goal 2**: is the proposal a stable feature selection system based on ensemble technique so as to yield stable results that are able to deal with different changes and alterations that may exists happen in data instances or/and attributes.
- **Research Goal 3**: is the proposal of sophisticated method meant to solve feature selection problem even in extreme cases where huge and complex interactions may exists in large datasets.

Contributions of the thesis

This thesis focuses mainly on feature selection research and its application in high dimensional classification datasets. The main contributions of this thesis, which are meant to study the aforesaid challenging problems and achieve the above noted research goals by developing new systems for feature selection, are briefly mentioned in the figure 4, and they are sequentially stated in the following list:

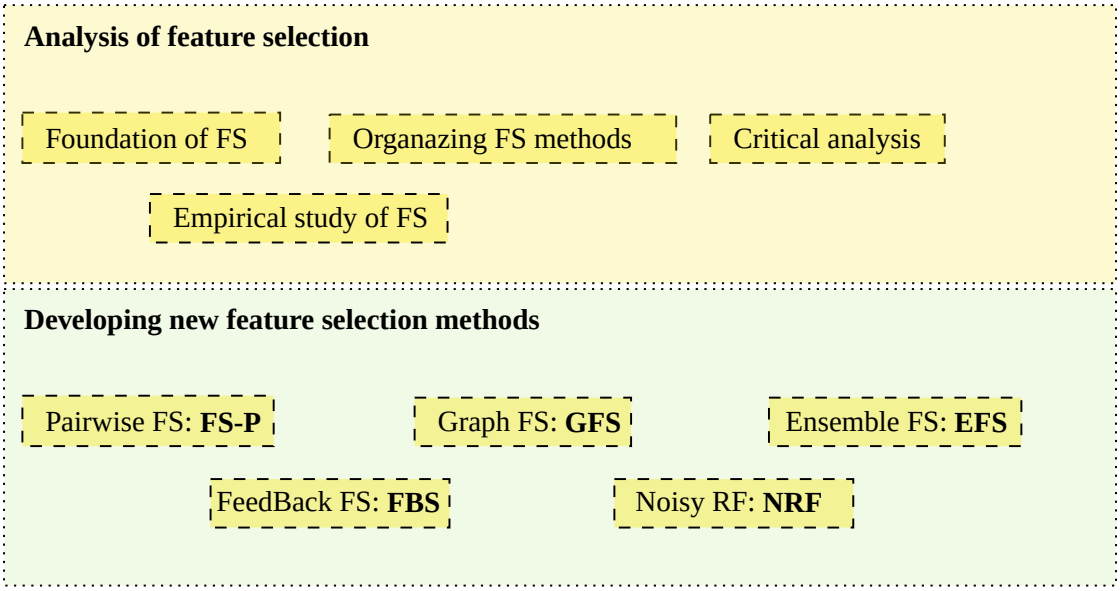


Figure 4 – Organization of the thesis.

1. Organizing the state-of-the-art of feature selection in order to provide a clear overview for the reader.
2. Providing a critical and empirical analysis of existing feature selection algorithms so as to supplement insights and recommendations to help researchers and enlighten readers.
3. Proposing new feature selection algorithms, which should be able to reduce the dimensionality of a given problem while maintaining or even increasing the performance of learning algorithms.
4. Creating a launching ground for the comparison of feature selection methods.

In this thesis, a stressed emphasis is placed on supervised feature selection and its main objectives which are separately detailed and instantiated in the following key-insights:

Analyzing the state of the art of feature selection methods.

We will organize the state of the art of feature selection with the purpose of constructing a clearly informative and holistic overview that can serve the reader. In addition, we will provide a critical review of the most popular feature selection methods by empirically evaluating their performance using several benchmarking datasets so that we can expose their advantages and drawbacks as well as their applicability. In addition, we will entirely overview feature selection technique.

Developing new feature selection methods

This second part of the thesis is devoted to propose novel feature selection methods that are efficient enough to identify and select the best subset of feature in high dimensional datasets. Five fundamental feature selection methods are proposed.

The **first contribution** is an intermixed patchwork of filter-wrapper feature selection methods based on pairwise features evaluation named ***FP-S***. Akhiat et al. (2017). The intermingled combination of feature selection suggested approaches benefits from the rapidity and the exploitation of filters' simplicity as well as the qualitative performance of wrappers.

The **second contribution** relies on graphs to represent features (variables). The central idea of this method is that the best features correspond to those nodes with the highest degree Akhiat et al. (2021a). A modularity function is applied to cluster the whole graph into communities and the best subset of features is selected among the best detected communities. This proposal shows its effectiveness in selecting the proper features compared with FS state of the art.

The **third contribution** consists of training different models using different classification algorithms Akhiat et al. (2019). Each model is trained using just one feature at a time. The set of all models represents our generated diverse library of models. Since each model corresponds to exactly one feature, the subset of selected models corresponds to the subset of optimal features.

The **fourth contribution** of this thesis work is the developpemnt of a new variant of random forest classifier which is named Noisy Random Forest ***NRF*** Akhiat et al. (2021c). The ***NRF*** provides unbiased variable selection where a noisy feature technique is used to address the baised ***RF*** selector. First, we add a noisy feature to each dataset. Second, the noisy feature plays the role of a stopping criterion. If the noisy feature is selected as the best splitting feature over other candidate ones, then we stop the creation process because at this level, the model starts to over-fit on the noisy features. Finally, the best subset of features is selected out of the best-ranked feature according to the Gini impurity of this ***NRF***.

Beyond the traditional view of feature selection formalization, the **fifth contribution** is inspired by the reinforcement learning approach where feature selection can be performed in a specially exceptional way. Feature selection can be seen

as a Markov decision process where a state represents each feature subset. The decision tree branch structure is used to represent each state and demonstrate the transition that occurs between them. Alongside the exploration of the state space, our system exploits the pre-gathered experiences using the proposed transition similarity measure (***TSM***). This system moves through the available state space and tries to discover new rules, gathers experiences and exploits what it has already experienced to find the best features subset.

Introducing a critical framework of feature selection methods comparison.

The first mind-boggling question facing practitioners or researchers once they embark on evaluating and performing feature selection methods is the following: what are the aspects we would like to check and evaluate in a given dataset? In this thesis, we decided to evaluate feature selection methods with the regard to the following aspects: 1) Area under the Curve (AUC) metric. 2) Stability of the selected subset. 3) The size of feature subset. 4) Relevance of the selected features.

Structure

Thesis chapters are organized in a ladder-like manner which is characterized by a logical interconnection and cohesive smoothness of the mainly existing and the newly proposed feature selection methods offering the reader an inclusive-whole glimpse of the quintessential key-points being highlighted in this thesis.

- The **chapter 1** introduces, motivates and surfaces feature selection problem. It also presents the main definitions and concepts that are indispensable to dive deep into feature selection details and to understand well the quintessential thesis that forms the central infrastructure of our research topic. In addition, the whole feature selection procedure is deconstructed and explained. First, it starts by shedding light on outstanding problematic challenges caused by the massive amount of data generated through many resources. Second, this chapter formalizes the feature selection problem in supervised tasks so as to provide readers and researchers with a basic springboard and preliminary groundwork as initial blueprints for the feature selection field.
- the **chapter 2** is dedicated to review the coverage of the most popular works that are developed and introduced in the sphere of feature selection, and investigate their performances through assessing their strengths and weaknesses. More succinctly, this chapter provides an insightful analysis that covers different feature selection methods in order to provide some insights and recommendations to readers and researchers.
- The **chapter 3** focuses mainly on the properties of the evaluation methodology commonly applied in feature selection. First, the evaluation methodology is well presented and discussed. Then, the real-world datasets employed to assess the performance of feature selection methods are described. In addition, the wiedly used evaluation metrics and techniques have been listed and discussed.

- The **chapter 4, 5, 6** and **7** focuse mainly on developing and proposing new feature selection frameworks and methods. Five feature selection methods are proposed in this thesis which are:
  1. In **chapter 4**, two novel feature selection methods are presented. The first proposal is a hybrid filter-wrapper approach. It selects the best features relying on a pairwise feature evaluation. In the second contribution, a graph based representation for feature selection method is proposed and discussed in this chapter. The reason behind the incorporation of a graph structure in feature selection is that graphs are inclusively holistic and powerful means of data representation that enable us to schematize complex real-world phenomena considering each feature as a node embedded within the graph. The best feature subset generated by this proposed method is represented by the nodes with a highest fitness value.
  2. Since the ensemble method is a machine learning technique that combines several weak models to reduce variance and improve prediction performance, in **chapter 5**, we propose an ensemble feature selection method which produces more stable results than a single selector. First, for each feature, we train a different model using different classifiers and different parameter settings. The set of models represents our diverse library. Second, we use a selection with replacement technique to find the optimal subset of models that when averaged together yield excellent performance and stable subsets.
  3. The **chapter 6** describes in details the new variant of RF algorithm we propose for feature selection. We first add a noisy feature to each used datasets as a generated feature. Second, the noisy feature is used as a stopping criterion during the construction of the new RF. Once the noisy feature is selected as the best splitting variable over other candidates, we stop the splitting process because, at this level, the model starts to over-fit on the generated noisy feature. Finally, the best subset of features is selected out of the best-ranked feature regarding the Gini impurity of the new developed RF version. The obtained results confirm that our algorithm provides unbiased and reliable selector compared with RF.
  4. Beyond the traditional formalization of the feature selection problem, we proposed a feature selection system in **chapter 7** which is informed by the reinforcement learning and Markov decision process as a point of departure to embark on a new feature selection method. Each subset of features can be seen as a state, which is represented by a decision tree branches.
- **Conclusion and Future Work**: As to the last chapter, we conclude the thesis by summarizing the main concepts, topics, findings, and the attained results. Moreover, we suggest some recommendations for researchers as well as providing some perspectives in relation to feature selection.
- **Publications of the Author**: This section lists the publications of the author produced during the preparation of this dissertation.

1

Feature Selection: Definitions and Concepts

“

"Data is the new science. Big data holds the answers."

Pat Gelsinger.

Chapter 1

Feature Selection: Definitions and Concepts

Contents

This first chapter presents an introduction to feature selection (FS) and its importance in machine learning area. Moreover, it describes the main definitions, terminologies and concepts that are mandatory before sinking deep into details.

1.1 Introduction

With the rapid development of novel technologies and applications, an immense amount of data is accumulated and generated from many resources and domains such as social media, bioinformatics, marketing, internet of things and biometrics etc (see figure 1.1). The accumulated data may not only be of high-dimensional in terms of data instances (samples), but also in terms of features (variables) which bring many challenges to machine learning algorithms. Therefore, a variety of

machine learning and data mining algorithms fail to scale on large size real-world problems. In the literature, The term high-dimensionality is applied to a data that includes one of the following aspects:

(a) The number of samples is very high

(b) The number of features is very high

(c) Both the number of samples and features are very high

(d) Refers to a dataset in which the number of features n is larger than the number of examples (instances) m, often written as $n \gg m$ Minasny (2009).

(e) The number of features times the number of observations is greater than 10000. More formally: $n \times m > 10000$ Zhao and Liu (2012).

There exists in the literature some debate about the term high-dimensionality since some authors claim that it only refers to the feature space whereas others use it vaguely for both features and observations. In this thesis work, a dataset will be deemed of very high dimensionality when the aspects (b), (c), (d) or (e) are presented in a dataset.

To efficiently manage a very high-dimensionality datasets for the sake of making it available and automatically extract insights, knowledge and hidden patterns, it is a necessity to apply dimensionality reduction tools and techniques.

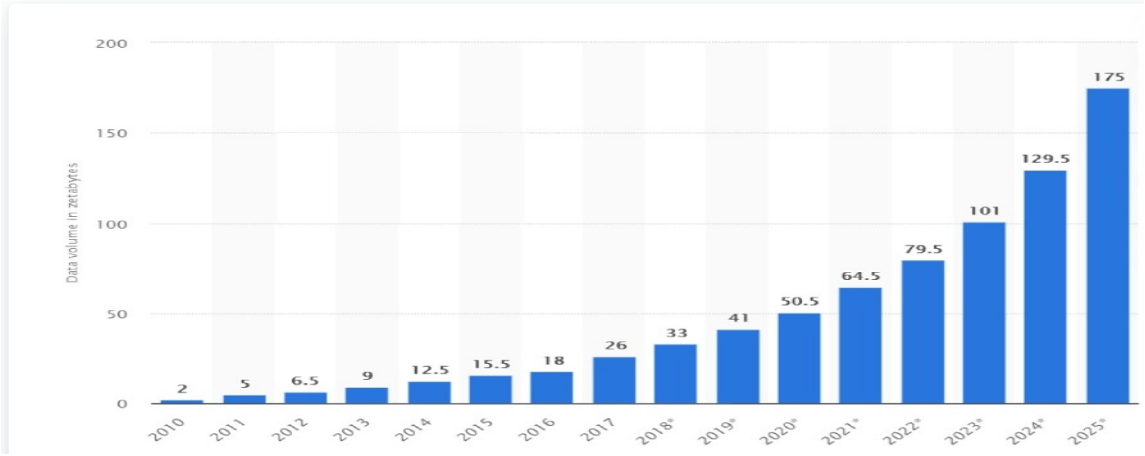


Figure 1.1 – Volume of data generated world-wide from 2010 to 2025 in Zetabytes. This statistics come from the Everis knowler: https://www.everisknowler.com/data-fabric-manage-fast-growing-data

1.2 Machine learning

In 1959, Arthur Samuel defined machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed". In contrast to traditional computing where algorithms are just a collection of explicitly programmed instructions, machine learning, as a progressive developing field,

is meant to study algorithms and techniques that allow computers to automatically "learn" from pre-gathered experiences (data) without being explicitly programmed or assisted by humans. Nowadays, any technology user has benefited from machine learning. ML is used anywhere from image recognition to speech recognition, recommendation systems, bioinformatics, social media to self-driving cars etc. Machine learning implementations are classified into: supervised; unsupervised, semi-supervised and reinforcement learning according to the nature and the availability of learning class label.

1.2.1 Supervised learning

Supervised learning is a machine learning task of modeling the relationship and dependencies between the input features X and the target class Y. More formally, in Supervised learning, we try to find a mapping (an inferred function) function f between X and Y such that the output of new data can be predicted based on the mapping function learned in the training phase: $Y = f(X)$. Supervised learning can be further grouped into regression and classification problems. We talk about classification problem when the class label Y is a category and about regression when the output Y is a real value. The popular algorithms of supervised learning include Support Vector Machine (SVM) Safavian and Landgrebe (1991), Logistic Regression (LR) Hosmer Jr et al. (2013), Decision Tree (DT) and Random Forest (RF) Breiman (2001).

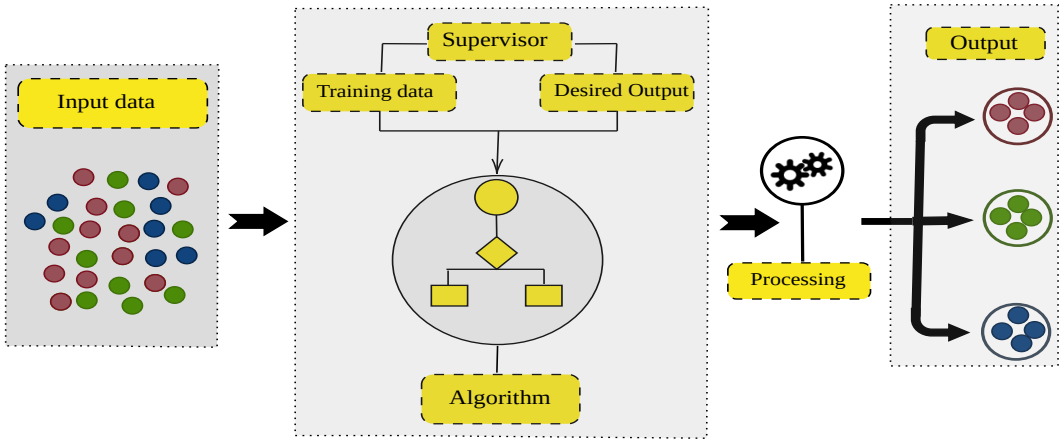


Figure 1.2 – Supervised learning

1.2.2 Unsupervised learning

As opposed to supervised learning, unsupervised learning algorithms are trained on unlabeled data where only the input data X is available. The main goal of unsupervised learning is to identify the hidden patterns within a dataset. Unsupervised learning methods decide which objects should be grouped together as one class. In other words, they learn classes by themselves. K-means Kanungo et al. (2002) and C-means Kanungo et al. (2002), anomaly detection methods Chandola et al. (2009) and Auto-encoder are widely used methods for unsupervised learning Ng et al. (2011).

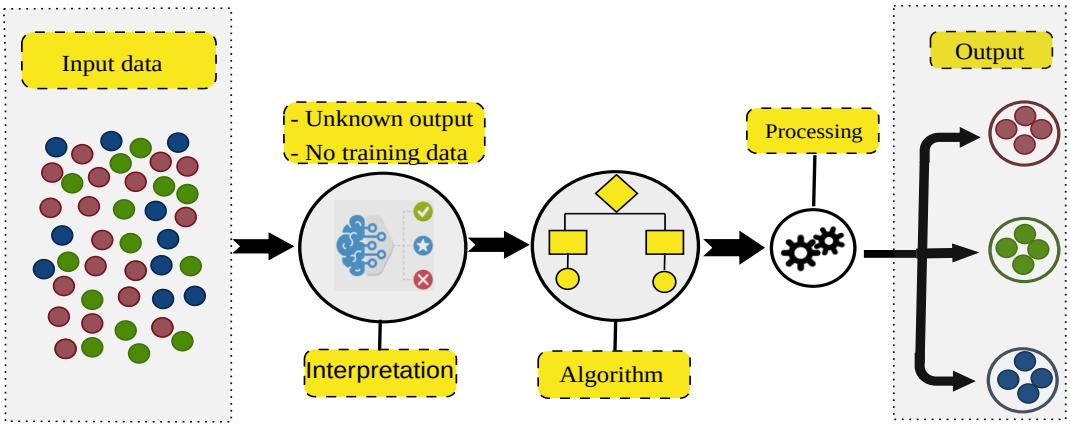


Figure 1.3 – Unsupervised learning

1.2.3 Semi-supervised learning

Semi-supervised learning is somewhere in between supervised and unsupervised learning as it combines both labeled and unlabeled data during the training process. Typically, a small amount of labeled data and a large amount of unlabeled data Zhu and Goldberg (2009).

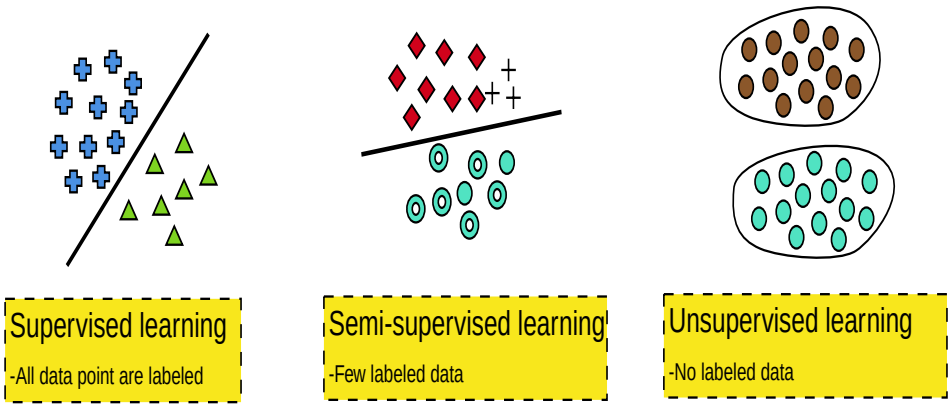


Figure 1.4 – The key differences between supervised, unsupervised and semi-supervised approaches. In Supervised learning, all data instances are labeled, for unsupervised learning, the target label is not provided while semi-supervised approach, the majority of data instances are not labeled and few of them are labeled.

1.2.4 Reinforcement learning

Reinforcement Learning (LR) Zhu and Goldberg (2009) is a type of unsupervised learning and the data provided to the system are unlabeled. It is a type of dynamic programming where the system is an intelligent agent capable of taking actions and interacting with its environment. As a result, the agent receives new state and reward. As iterations take place, based on the reward, the agent behavior becomes improved. Figure 1.5 describes the general overview of RL.

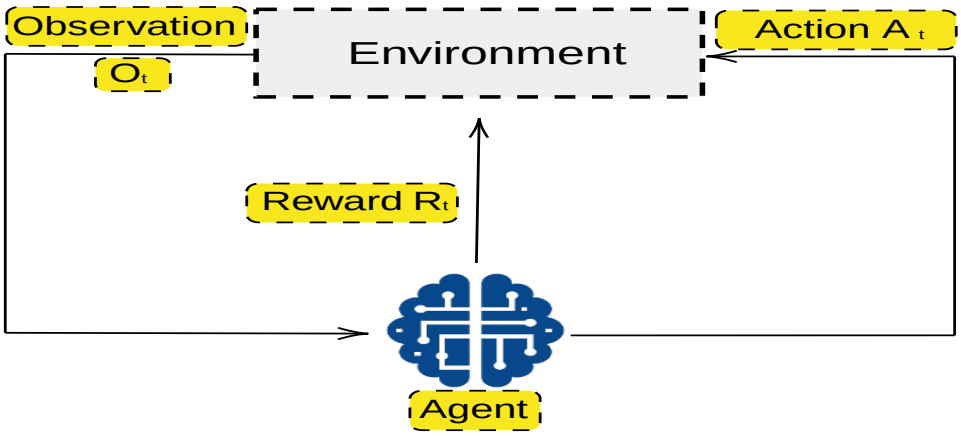


Figure 1.5 – Reinforcement learning paradigm.

1.3 Dimensionality reduction

Feeding high dimensional data to machine learning models without any pre-processing, generally leads to unsatisfactory learning performance due to the problem known as "curse of dimensionality" Friedman et al. (2001). This critical problem refers to many phenomena that arise when data become sparse in high dimensional spaces which may lead models to over-fit on training data Tang et al. (2014). Therefore, finding a reduced data matrices representation that can efficiently summarize the original ones before model induction should be considered. This latter process is known in the machine learning field as dimensionality reduction technique. Dimensionality reduction, which is the most powerful pre-processing tool, is divided into: feature extraction and feature selection. Both Feature selection and extraction are essential techniques to deal with the massive amount of data produced through many resources.

- Feature extraction: This technique achieves dimensionality reduction by transforming the original features into a reduced space, which results in generating a set of new features. The projection of original features could be either linear or non-linear according to the discrimination of the data instances. The generated set of features is usually characterized by its ability to discriminate class label. In some applications such as signal processing and information retrieval in which the most important aspect is the performance of the model, it is preferable to apply feature extraction. Some well-known feature extraction methods are the following: Multi-dimensional scaling Cox and Cox (2008), Isomap Tenenbaum et al. (2000), Local linear Embedding Roweis and Saul (2000), Principal Component Analysis Barshan et al. (2011) and to mention but a few Khalid et al. (2014); Trier et al. (1996); Ding et al. (2012); Preece et al. (2008); Guyon et al. (2008a).
- However, when model interpretability and simplicity are mandatory concerns, Feature selection would be a great alternative (choice) because it selects the most important features from the original set and removes the redundant and irrelevant ones without any transformation. In fact, the selected subset of features using feature selection approach maintains the

physical meaning of the original features which can be further used to successfully interpret the research domain Zhao and Liu (2007); Miao and Niu (2016).

1.4 Feature selection

Feature selection or variable selection is an essential research topic in the area of machine learning. As previously illustrated in section 1.2.1, according to the availability of the class label, feature selection can be classified into supervisedWeston et al. (2003), semi-supervised Xu et al. (2010); Li et al. (2017), and unsupervised feature selection Dy and Brodley (2004); Mitra et al. (2002). If all instances in the dataset are labeled (have known response variable) then, the process is called supervised feature selection. If there is only an exclusive number of instances that are labeled while others are not, in this case, it is obvious that we are referring to semi-supervised feature selection Sheikhpour et al. (2017). If none of the data instances is labeled, the process is deemed as unsupervised feature selection. Since the majority of research papers tackling the feature selection problem are in the supervised feature selection paradigm, this thesis will focus mainly on supervised feature selection in the classification problems. In order to justify that, a literature search using the keywords "feature selection classification", "feature selection regression", "Supervised feature selection" and "Unsupervised feature selection" has been conducted of publications listed at Scopus during the last 10 years (from 2010 to 2020). The obtained results are presented in the figures 1.6 and 1.7 which clearly justify that fact.

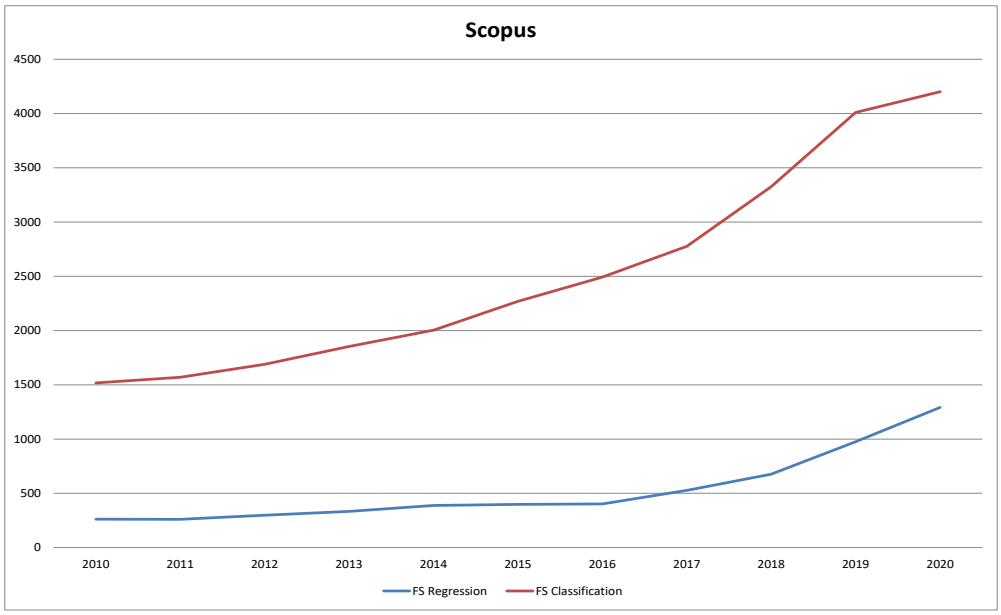


Figure 1.6 – A Comparison of Published Feature Selection Studies for Classification and Regression on Scopus.

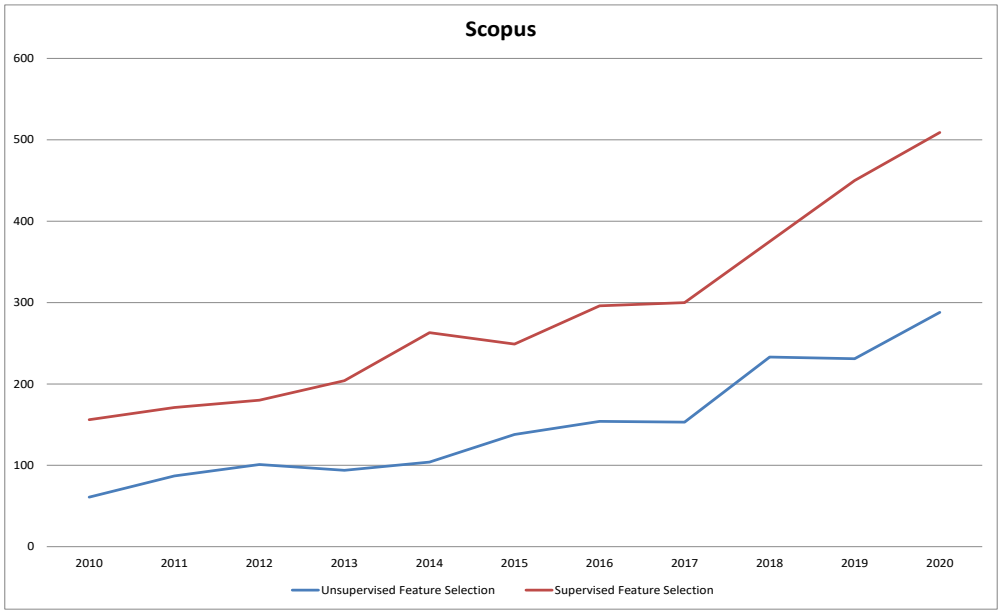


Figure 1.7 – A Comparison of Published Feature Selection Studies for supervised and unsupervised category.

Feature selection (also known as subset selection or variable selection) is a pre-processing step in machine learning meant to deal with the high dimensionality problem. It can be defined as the process of selecting the relevant features and removing the irrelevant, redundant and noisy ones (see Figure 1.8), intending to obtain the best performing subset of original features without any transformation. FS is a challenging research topic for several domains, including Deep Learning, Pattern Recognition, Machine Learning and Information retrieval. FS can serve several purposes Chandrashekar and Sahin (2014); Yadav and Shukla (2016); Molina et al. (2002a), such as:

- **Improving the performance of machine learning algorithms**
  Removing noisy and irrelevant features can result in less misleading instances. Thus, the model accuracy improves by implication. For example, the predictions of instance-based algorithms Aha et al. (1991) such as K-NN Zhang (2016), K-means Kanungo et al. (2002); Likas et al. (2003) may be drastically deviated by noisy features since they rely on a small neighborhood in the attribute space Wilson and Martinez (2000) and Maglogiannis (2007).
- **Improving generalization and learning speed**
  Keeping the redundant and irrelevant features in datasets can lead machine learning models to over-fitting problem Guyon and Elisseeff (2003). Removing those un-informative features first before evaluating machine learning algorithms may increase the generalization ability of models. Moreover, feature selection is able to reduce the dimensionality and complexity of the predictive models which eventually help to speed up the learning process.

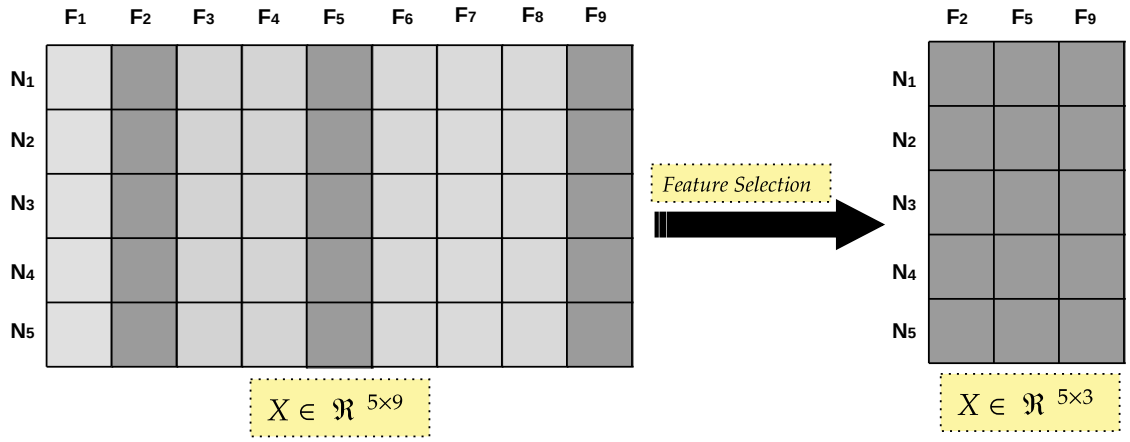


Figure 1.8 – Reducing the feature space $X \in R^{5\times9}$ by selecting the most informative features while maintaining the physical meaning of the original features. The new feature space is $X \in R^{5\times3}$.

- **Enhancing model interpretability**
  Given the presence of thousand of features in real-world datasets, it is hard to understand and interpret the modern learning algorithms after they have been trained as they are more complex Mangal and Holm (2018).

1.5 Feature selection formalization

Feature selection problem is wide-ranging in inductive machine learning and data mining setting and its importance is beyond doubt. Let X be the entire set of original features, with the cardinality: $|X| = n$ where $n \in N$. *Let* $F_k = \{ f_i \mid i \in \{1, 2, .., n\}\}$ represents a non-empty feature subset of size k where $k \ll n$. Let $\Psi : F \rightarrow \mathbb{R}$ a variable ranking function to be optimized (discriminatory ability of feature subset) where $F = \{F_k\}$. By convention, we assume that the more the score yielded by Ψ is higher, the more useful information the features can carry. Assuming that the optimal selected subset by the evaluation functionΨ is the subset S, then:

$$\Psi(S) = \max_{Fi \in F}(\Psi(F_i)) \tag{1.1}$$

FS can be defined as the process of choosing the best subset of features (relevant features) among the competing 2^n candidate subsets (n is the number of features) while maintaining or even increasing the model performance. Ideally, feature selection should search all 2^n feature subsets to find an optimal one regarding to some evaluation functions Ψ. This exhaustive search method cannot be even practically applied to datasets of medium size features set (medium n). The complexity of this intractable search strategy is exponential $O(2^n)$ in terms of size data Liu and Motoda (2007); Ang et al. (2015). Therefore, various heuristics search strategy are proposed and developed to deal with this problem Liu and Motoda (2007).

In this thesis, the ranking function Ψ should be maximized. Regarding to the task to be solved, Feature selection problem can be formulated with many different ways Kudo and Sklansky (2000); Siedlecki and Sklansky (1993). The main formulations of FS process can be seen under three considerations:

1. Seek the subset that yields the highest score of Ψ. Mathematically: find $F \subseteq X$, such that Ψ(F) is maximum where $|F| = k < n$.
2. Find the smallest$(k << n)$ subset of features for which the performance does not deteriorate below a given threshold. Mathematically: given a threshold Ψ_0 find $F \subseteq X$, where $|F|$ is very small, such that $\Psi(F) \geq \Psi_0$.
3. Find a compromise (trade-off) between minimizing $|F|$and maximizing $\Psi(F)$.

It is obvious that (1) is unconstrained optimization problem while (2) and (3) are constrained by extra conditions. With these three considerations, we should keep in mind that the optimal subset is not unique Belanche and González (2011).

Table.1 describes more precisely the structure of datasets that have been used in the context of supervised feature selection where a set of attributes$(f_1, f_2, .., f_n)$ describes the context and a set of classes$(C_1, C_2, \ldots, C_k)$ represents the target by a set of features.

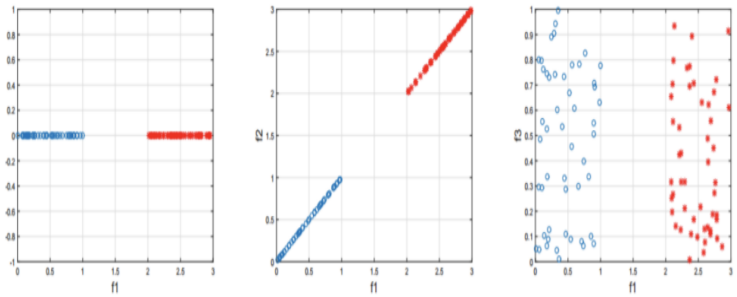
Table 1.1
Datasets structure for supervised feature selection.

	$\mathbf{F}_1$	$\mathbf{F}_2$	.....	.....	$\mathbf{F}_N$	**Classe**
Observation$_1$	*Value*$_{11}$	*Value*$_{12}$	.....	.....	*Value*$_{1N}$	C_1
Observation$_2$	*Value*$_{21}$	*Value*$_{22}$	.....	.....	*Value*$_{2N}$	C_2
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
Observation$_m$	*Value*$_{m1}$	*Value*$_{m2}$	.....	.....	*Value*$_{mN}$	C_k

1.6 FS concepts and definitions

After defining the feature selection problem more formally, let's invoke and review some frequently used concepts and definitions suggested in the literature. Usually, the set of original feature space consists of relevant, irrelevant, and redundant features as shown in Figure 1.9.

It is difficult to assess the usefulness and informativeness of features since the large number of the existing combinations of features contains diverse amount of information that could be used for data classification. Each dataset may contain irrelevant, relevant and redundant features. There are various definitions in the FS literature for what it means for feature to be relevant. Generally, the feature is



(a) relevant feature f_1 (b) redundant feature f_2 (c) irrelevant feature f_3

Figure 1.9 – For example, for a binary classification task (f_1 is relevant; f_2 is redundant given f_1; f_3 is irrelevant).

said to be relevant to the target concept if twiddling some instances of that feature drastically affects the classification given by the target concept Blum and Langley (1997). In contrast to relevant features, irrelevant features do not provide any useful information at all. The presence of these redundant and irrelevant features negatively affects the classification performance. In real-world datasets, many scenarios arising which may increase the complexity of feature selection problem. Some datasets may contain non-linear dependencies either between features or between features and target concept Y John et al. (1994) or interacting features that appear irrelevant individually but when collectively combined together they may be useful to the classification problem. For further illustration, let's consider the well-known XOR problem. The target concept (Y) is labeled as$\{+, -\}$ for simplicity. For XOR problem, there are two features F_1 and F_2 and the class label Y. The class Y is zero when the two features F_1 and F_2 have the same value, and Y is one otherwise. It is clearly that using just one feature to determine the class is impossible. Therefore, F_1 and F_2 are irrelevant when they are used separately and informative when combined together Blum and Langley (1997). Finally, as a recommendation for readers, practitioners and researchers, a careful selection should be carried out when we would like to perform feature selection especially when interactions between features are indispensable.

1.6.1 Feature relevance

Generally, a feature is considered as relevant if it provides some information about the target. John et al. (1994) suggested two degrees of relevance: weak relevance and strong relevance.

Let X be the set of all features, f a single feature and $F_i = X - \{f\}$. $P(.)$ denoted the distribution probability.

Definition 1 (Strong relevance)

A feature f is deemed to be strongly relevant if the removal of f alone from the feature space results in performance deterioration. More formally, a feature f is strongly relevant if:

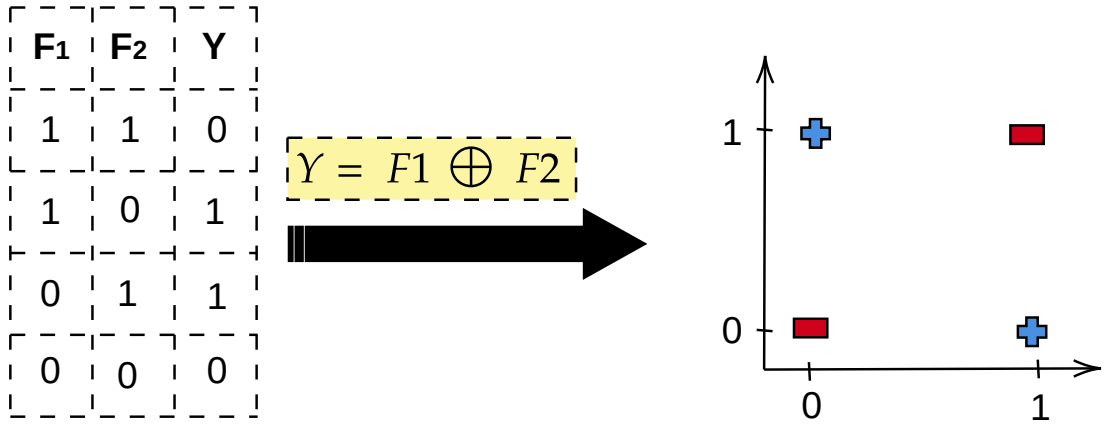


Figure 1.10 – XOR problem, a) is the XOR table of truth. b) $\bigoplus$ denotes the XOR. c) Scatter plot of XOR data into two-dimension subspace

$$P(C|f \cup F_i) \neq P(C|F_i) \; \forall F_i \subset X \tag{1.2}$$

Definition 2 (Weak relevance)

A feature f is weakly relevant if it is not strongly relevant, and there exist a subset of features S where the performance using $S \cup f$ is better than using just f . More formally, a feature f is weakly relevant if :

$$P(C|f \cup F_i) = P(C|F_i), \text{ and } \exists S \subset F_i, \text{ such that} P(C|f \cup S) \neq P(C|S) \tag{1.3}$$

Definition 3 (Irrelevance)

A feature f is considered irrelevant if it is not weakly relevant and not strongly relevant, Otherwise it is deemed irrelevant. This means that the removal of the irrelevant feature does not result in any performance deterioration. More formally, a feature F_i is irrelevant if:

$$\forall S \subset F_i P(C|f \cup S) = P(C|S) \tag{1.4}$$

1.6.2 Feature redundancy

According to Yu and Liu (2004a), feature redundancy notions are normally in terms of feature correlation. It is commonly accepted that if two features are correlated then, they might be seen as redundant. Nevertheless, the feature redundancy relies on the metric used for correlation evaluation (i.e. linear, non-linear correlation) and also, on their respective dependency to the target outcome. As opposed to the several studies that consider the correlation as a sufficient metric for redundancy detection Guyon et al. (2008a) , the recent ones put more emphasis on both feature correlation and class dependency Estévez et al. (2009) which is linked to Markov Blankets Koller and Sahami (1996).

Definition 4 (Markov blanket)

In machine learning, usually a subset of features is enough to derive and infer random feature or variable with a set of variables. The subset that contains all useful information is called "Markov blanket" Kyburg Jr (1991). More formally, given a feature $f \in F$, let $M \subset F$, such that $f \notin M$, the feature subset M is said to be a Markov blanket for f_i if:

$$P(F \backslash M\{f\}, (C|f \cup M)) = P(F \backslash M\{f\}, C|M) \tag{1.5}$$

Definition 5 (Redundant)

Let F be the current set of features, a feature is redundant and hence, it should be removed from F if it is weakly relevant and has a Markov blanket M within F. According to Yu and Liu (2004b), the whole feature set F could be partitioned into four distinct parts as follows: I) Weakly relevant and redundant features. II) Irrelevant features. III) Weakly relevant but non-redundant. IV) Strongly relevant features. The Figure 1.11 gives an overview of the feature selection and the relationship between feature relevance and feature redundancy.

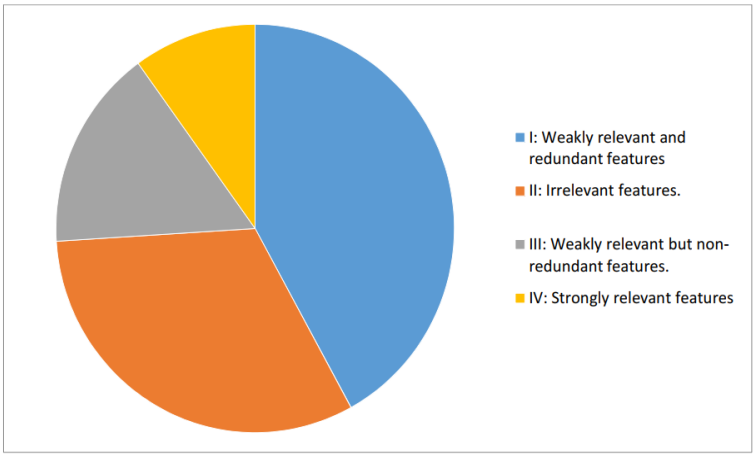


Figure 1.11 – Overview of feature relevance and feature redundancy.

An optimal subset of feature should contain the strongly relevant features, a subset of weakly relevant and none of irrelevant features (III+IV: optimal subset see 1.11).

1.6.3 Feature interactions

In the literature, it has been shown that identifying the relevance of features separately without taking into account the interaction among them may not be that hard. However, the individually irrelevant features may become weakly or strongly relevant when interacted and combined with other features which makes features removal a very challenging task (see the XOR example in 1.10). In Lavrač et al. (2003), Jakulin and Bratko proposed the interactions among subsets of retained features. Indeed, a feature might drop its relevance due to the absence of interacting features.

1.7 Feature selection process

Generally, feature selection process consists of two phases, search and validation. The search phase can be divided into three steps. In this section, we describe in detail the four main steps (subset generation, subset evaluation, stopping criterion and validation) as shown in Figure 1.12.

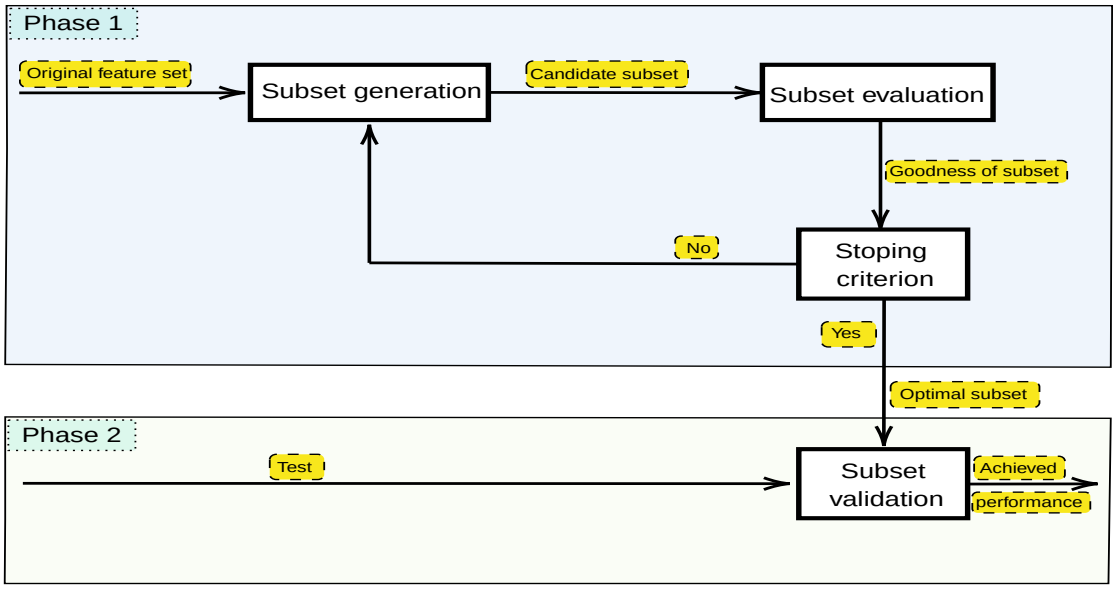


Figure 1.12 – Feature selection process.

1.7.1 Phase 1: search

1.7.1.1 Subset generation

Subset generation is an important process of heuristic search, where each state in the search space represents a candidate subset for evaluation. At the beginning of the process, the full original feature space is required as input of the feature selection process, and as a result, the FS procedure produces the best feature subset as well as its performance attainability. Moreover, the nature of this process is determined by two basic points: first, one must decide the starting search strategy. Various starting search strategies are proposed in the literature. The search may start with an empty subset and successively adds features (forward), a full subset and successively removes features (backward), a randomly selected subset or simultaneously adds or removes features (stepwise or bidirectional). The search starting point is very important as it largely influences the search direction. Second, one must decide which search strategy they should follow. For datasets with n features, 2^n possible feature subsets which make the space search exponentially intractable for evaluating all subsets (exhaustive search) even for datasets of medium size (Moderate n). Various search strategies have been introduced such as: complete search (exhaustive), sequential search and random search.

- **Complete search** The complete space exploration, which grows combinatorially, is the only way that guarantees the selection of the optimal subset with the regard to a given criterion function. Since the complexity of

the complete search is $O(2^N)$, the search should not be exhausted and some heuristic search method should be used . Branch and bound is a more realistic search strategy introduced in Narendra and Fukunaga (1977). It identifies the optimal feature subset only if the used criterion function is monotonic. Moreover, if the number of features is more than 30, the search becomes infeasible and intractable Ang et al. (2015).

- **Sequential search**
  An exhaustive search examines and evaluates every subset in the search space; however, a large amount of computation is required. Many fast algorithms have thus been proposed to reduce the required computation at the price of slightly dropped performance, such as sequential forward selection, sequential backward elimination and stepwise selection. Moreover, in the sequential search, the optimal subset is not guaranteed but its algorithms are simple to implement and fast in producing results as the order of the search space is usually $O(N^2)$ or less.
- **Random search**
  The idea underlying this type of search is to use its randomness which helps to escape local optima in the search space Belanche and González (2011) that many algorithms might get stuck in (for example greedy hill-climbing Caruana and Freitag (1994)).

1.7.1.2 Subset evaluation

Each generated subset in the previous step needs to be evaluated using a criterion function. Various evaluation criteria are used to assess the effectiveness of the generated subset. Therefore, an optimal subset selected by one criterion would not be necessarily an optimal one selected by some other evaluation criteria. The widely used evaluation criteria can be categorized according to the dependency on algorithms into independent and dependent criteria. The independent criteria are usually used for filter methods. They evaluate the excellency of the generated subset relying on some intrinsic characteristics of the data without involving any algorithm. There is an eclectic variety of independent criteria among of which are these prominent ones: distance measures, information measures and consistency measures, to mention but a few Aha et al. (1991). As opposed to independent criteria, dependent criteria rely on the performance of learning algorithms to determine which feature should be kept and which one should be discarded.

1.7.1.3 Stopping criteria

A stopping criterion is a deciding stage used to stop the feature selection process. The mostly predominant and widely employed stopping criteria are the following:

a) The search completes.
b) Some given conditions are satisfied (minimum number of features or maximum number of iterations).
c) Addition (or deletion) of any feature does not produce a better subset.
d) A sufficiently good subset is identified.

1.7.2 Phase 2: Validation(Subset validation)

The robustness and the stability of the best selected subset from the first phase are assessed in the second phase. This latter is usually separated from the first phase (search process) to decrease the risk of a non biased selection process. Generally, when the prior-knowledge is available (irrelevant and relevant features in synthetic datasets), the best selected subset is evaluated on it because the irrelevant and relevant features are known beforehand. In real-world datasets such information is not always available. Therefore, we have to rely on some indirect methods by controlling the change of mining algorithms' performance with regard to the change of features.

1.8 Summary of the Chapter

Through this chapter, we have presented the main definitions and concepts that are indispensable and mandatory to dive deep into feature selection details and to understand well the quintessential thesis that forms the central infrastructure of our research topic. Moreover, a general procedure of feature selection is thoroughly detailed and foregrounded to the forefront. In the first part, we have started by shedding light on outstanding problematic challenges engendered by the massive amount of data generated through many resources. Then, as a practical alternative, we have tackled the serviceability of the dimensionality reduction tools which are generally categorized into: Feature selection and feature extraction. As our main focus is on feature selection category, we have briefly overviewed the widely known feature extraction methods. On the other side, we have formalized the feature selection problem in supervised context by deconstructing and detailing the important definitions and concepts as well as the whole feature selection framework providing readers and researchers with a basic springboard and preliminary groundwork as initial blueprints for the feature selection field.

In the next chapter 2, we will discuss and describe the categorization of feature selection methods (Filter, Wrapper and Embedded) and we will provide a critical analysis and comparison of each category (advantages and disadvantages). Moreover, a pseudo-code of the most popular methods will be provided. In addition, empirical comparisons are conducted to assess and evaluate the performance of nine FS methods and their applicability as well. Finally, guidelines for applying features selection are cited and discussed.

2

Feature selection methods

> "Too much data is not always good for machine learning algorithms. Sometimes too much data could be misleading.
>
> **Yassine Akhiat.**

Chapter 2

Feature selection methods

Contents

While the first **Chapter 1** is dedicated to cast light on the most important definitions, concepts, terminologies and feature selection, **chapter 2** reviews the most popular feature selection methods in the literature by empirically evaluating, critically analyzing and empirically assessing their performance as well as summarizing their prominent drawbacks and advantages. As to the end of this chapter, some recommendations and crucial guidelines for properly applying feature selection methods either by researchers or practitioners are provided.

2.1 Introduction

Feature selection has been widely studied in the past few years by machine learning researchers and specialists as it is a very important pre-processing before model induction. FS has been successfully applied in various domains from text categorization Forman et al. (2003); Gomez et al. (2012) to, DNA microarray analysis Yu and Liu (2004c), bioinformatics Saeys et al. (2007) to image recognition and music retrieval Jain and Zongker (1997a); Dy et al. (2003). Because feature selection has been a dynamically ever-evolving field of research for decades, innumerable research papers and books have been published in the literature Stańczyk and Jain (2015); Guyon et al. (2008b); Ferreira and Figueiredo (2012). However, nobody can claim the permanence or the everlasting superiority of a powerful feature selection method" Bolón-Canedo et al. (2013). Therefore, one should find the appropriate feature selection method for a specific problem. There is a variety of strategies that could be used to introduce and propose new FS method:

- Combining several FS methods either from the same category or from different categories (filter and wrapper) such as our proposed pairwise feature selection introduced in Akhiat et al. (2017).
- Generating inspiration from other domains and incorporate them for the sake of solving feature selection problem differently such as the proposed feedback feature selection which is inspired by the reinforcement learning Sun and Li (2006); Akhiat et al. (2021b).
- Integrating FS methods with other algorithms Saeys et al. (2008).
- Aggregating an ensemble of FS methods to enhance the overall performance of the final selected features Bolón-Canedo et al. (2014); Molina et al. (2002b).

According to Yu and Liu (2004a), feature selection methods can be divided into: individual evaluation also known as feature ranking Guyon and Elisseeff (2003), and subset evaluation. In individual features evaluation, features are ranked according to their importance. Features with the highest importance are the best at distinguishing instances from different classes. This approach is efficient dealing with high dimensional datasets as it ignores the interaction between features. Moreover, individual evaluation fails when dealing with redundancy on the ground that some features can have equal similarity in ranking. In the other hand, subset evaluation tries to select the optimal subset that satisfies some criteria. Since subset evaluation takes into consideration the underlying interactions

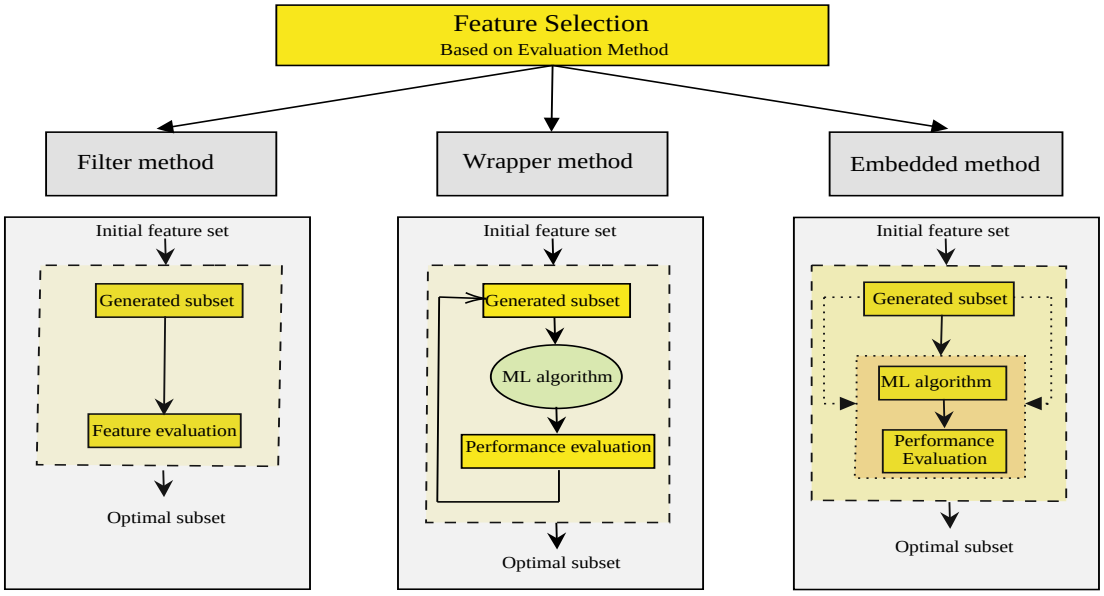


Figure 2.1 – The broadly categorization of feature selection methods based on evaluation criterion.

between features, it is able to handle feature redundancy with feature relevance. Aside from this classification, Feature selection methods can be further categorized into three main approaches: Filter, wrapper and embedded approach (See Figure 2.1).

- **Filters**, which rely on some underlying feature characteristics of training data and mainly on its relationship with the class label to perform feature selection without involving any modeling algorithm. This approach is advantageous in terms of speed and generalization ability.
- **Wrappers**: features are evaluated together not separately as in filters. They rely on learning algorithms to assess and evaluate the relative performance of each generated subset. Precisely, this leads to make the learning process very expensive in terms of models complexity. However, this interaction with the learning model prone to produce better performance than filters.
- **Embedded**: This approach is able to identify the dependencies at a reduced computation complexity than wrapper as feature selection is performed and optimized within the classification process.

2.2 Filter methods

Filters rely on intrinsic characteristics of data to assess feature relevance. This category is a pre-processing step, which is independent of the induction algorithm as illustrated in Figure 2.1. Filters consist of two main steps. First, it ranks features individually based on a specific criteria measure like distance, Pearson correlation, and entropy, to mention but a few Zhao et al. (2010). Second, it selects the best-ranked features using a threshold value to induce the classification model. The remaining features are deemed to be irrelevant and useless. Afterwards, the

selected subset of features is presented as input of the induction classifier. Filter methods are fast, which makes them useful for high dimensional data. As the interactions between the independent variables are not taken into account, redundant features are allowed to be selected. Some of the most popular filter methods will be described in the following sub-subsections.

2.2.1 Information gain

Information Gain (IG) is an entropy-based evaluation method and a univariate feature selection method as well. IG allows to understand how much information each feature carry about the target concept. However, IG can also be defined with mutual information. In particular, information gain IG (F) is the reduction in the entropy that is archived by learning a feature F:

$$IG(F) = H(S) - \sum_{i} \frac{S_i}{S} H(S_i) \quad (2.1)$$

Where $H(S) = -\sum_{i=0}^{n}[p(i)logp(i)\ ]$ is the entropy of a given dataset S and $H(S_i)$ is the entropy of the ith subset produced by splitting S based on feature F. In feature selection context, features are ranked according to their IG score separately. Generally, the feature with high information gain should be ranked higher than other features since it has a powerful ability to discriminate data from different classes Quinlan (1986).

2.2.2 Relief

Kira and Rendel Kononenko (1994) in 1992 proposed a method called "Relief", which is an instance-based method. It tries to find for every training instance, its nearest neighbor from the same class (nearest hit), and from the opposite class (nearest miss). The score of each single feature is the difference between them. Relief calculates a feature score W_i for each feature which can then be applied to rank and select the top scoring features for feature selection. Since the weighting W_i of each feature is iteratively updated with each selected instance, Relief method can be highly efficient for high dimensional datasets. Feature weight decreases if it differs from that feature in nearby instances of the same class more than nearby instances of the other class and increases otherwise. The weight of each feature is computed as follows:

$$W_i = W_i - (x_i - nearHit_i)^2 + (x_i - nearMiss_i)^2 \quad (2.2)$$

The limitations of this algorithm are that it is applicable only to two-class classification problems and fails to handle redundancy. To fix the mentioned problems of Relief, an extension to Relief called Relief-A is proposed in Yu and Liu (2004a) for addressing the incomplete data problem. To address the multi-class problems, Relief-F is introduced Robnik-Šikonja and Kononenko (2003). The applicability of this extension is equally tantamount to Relief except that it has been adapted to be

generalized to multi-class problem. For given randomly selected instances, Relief-F method tries to find the k nearest neighbors from the same class (nearest hit) and from the opposite classes (nearest misses). Therefore, the quality of features is updated according to their powerful efficiency in making proper separation of the examples from different classes.

2.2.3 Fisher score

Fisher is an instance-based method; it considers that features with high quality should assign similar values to instances from the same class and different values to instances from different classes. With this intuition, the i_{th} feature F_i will be computed using the following formula:

$$F_i = \frac{\sum_{j=1}^{C} n_j(\mu_{ij} - \mu_i)^2}{\sum_{j=1}^{C} n_j \sigma_{ij}^2} \tag{2.3}$$

Where μ_{ij} and σ_{ij} are the mean and the variance of the i_{th} feature in the j_{th} class, respectively, n_j is the number of instances in the j_{th} class, and μ_i is the mean of the i_{th} feature. As Fisher Score assesses features individually, it cannot handle feature redundancy. The authors of Gu et al. (2012) proposed new variation of Fisher score called "a generalized Fisher score for feature selection". It jointly selects features, which aims to find a subset of features that maximizes the lower bound of traditional Fisher score.

2.2.4 Chi-square

Chi-square (Chi-2) is a statistic test of variables' independence Jin et al. (2006). For feature selection, the Chi-square technique computes the dependency between each feature and the associated class label relying on the following formula:

$$x^2 = \sum \frac{(Y_0 - Y_E)^2}{Y_E} \tag{2.4}$$

Where Y_0 is the true value and Y_E is the expected value. When Y_0 and Y_E are very close which means the two variables are independent, thus, the obtained x^2 value is smaller. The higher x^2 value indicates that the feature is more dependent to the class label which means that it is useful at discriminating instances from different classes. Chi-square limitations include its sample size requirements, the hardness of interpretation in case of the presence of a large number of categories in the independent or dependent variables McHugh (2013).

2.2.5 Mutual information

This method indicates the amount of shared information between a feature and the class label (between two features). In other word, MI is the measure of the amount of one random variable has about the other Gu et al. (2012). The best feature is the one that have a lot of shared information with the target class, which

means it can distinguish efficiently the members of one class from another. Mathematically, MI is defined as follows:

$$I(X,Y) = \sum_{i=1}^{n}\sum_{j=1}^{n} p(X(i),Y(j)).\log(\frac{p(X(i),Y(j))}{p(X(i)).p(Y(j)}) \tag{2.5}$$

MI is zero when X and Y are statistically independent $(p(X(i),Y(j)) = p(X(i).p(Y(j))$.

2.2.6 Minimum Redundancy Maximum relevance (mRmR)

In 2005, Hanchuan Peng proposed a heuristic Minimum-Redundancy-Maximum Relevance (mRmR) algorithm for feature selection problem. The mRmR tends to select features with a high correlation with the target class (output) and a low correlation between themselves. mRmR tends to minimize redundancy, and uses a collection of intuitive measures of relevance and redundancy to identify the best features for both continuous and discrete datasets. An improved method is proposed in 2013 by Mandal and Mukhopadhyay for gene expression Mandal and Mukhopadhyay (2013).

2.2.7 Pearson

Pearson method is a correlation based method; it provides a ranking weight between -1 and +1 to indicate the extent to which two features are linearly related. If the weighting is closer to zero this means that there is no linear correlation between the two features, otherwise, either features are strongly correlated positively (if +1) or negatively (if -1). Pearson can be very efficient to select relevant features when the correlation between features and the target class is linear, but usually in real-world datasets, the correlation is not the case. The next equation is used to calculate the PC between the independent variable X and dependent variable Y:

$$PC(X,Y) = \frac{\sum_i (X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_i (X_i - \overline{X})^2 (Y_i - \overline{Y})^2}} \tag{2.6}$$

2.2.8 Correlation-Based Feature Selection (CFS)

CFS Hall and Smith (1998) is a simple multivariate filter algorithm that evaluates the goodness of feature subsets by considering the individual predictive ability of each feature along with the degree of redundancy between them. The main idea of CFS is to select subsets that contain features that are highly correlated with the class and uncorrelated with each other. Irrelevant features are to be eliminated because they will have low correlation with the class. Redundant features should be discarded as they will be highly correlated with the remaining features.

2.2.9 Statistical dependence Measure ($\mathcal{M}_d$)

The $\mathcal{M}_d$ filter Bolón-Canedo et al. (2011) is an extended version of mRmR. Instead of using mutual information MI, It uses a measure of monotony to evaluate

feature relevance. The main idea behind $\mathcal{M}_d$ is the inclusion of the parameter λ that controls the relative emphasis spotlighted on relevance and redundancy. For $\lambda = 0$, the effect of the redundancy is vanished and the measure is relied only on maximizing the relevance. On the other hand, when $\lambda = 1$, it is more important to minimize the redundancy among variables. Seth and Principe (2010) stated that $\lambda = 1$ performs better than other λ values. In the context of classification where Y is a discrete variable we can formally write:

$$\mathcal{M}_d(X,Y) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\int (P(Y=y_j,\ X\leq x_i) - P(Y=y_j)P(X\leq x_i))^2 \qquad (2.7)$$

In brief, the feature selection algorithm can be described as follows, given a set of features $F_i \subset X$, choose f_i in the following way:

$$f_i = argmin_{X_i \in X/F_i}\left(\mathcal{M}(X_i,Y) - \frac{\lambda}{|F_i|}\sum_{X_i\in F_i}\mathcal{M}(X_i,Y_j)\right) \qquad (2.8)$$

where Y is the target and λ is a free parameter that controls the relative emphasis given on the relevance and the redundancy. Here, the relevance is evaluated by the dependence between a variable and the target, whereas the redundancy is evaluated by the average dependence between the new variable and the already selected variable.

2.2.10 Efficient Correlation Measure Based Filter (ECMBF)

ECMBF Jiang and Wang (2016) algorithm makes use of correlation measure between continuous and discrete features (CMCD) to measure the correlation between a continuous and a discrete feature, and choose linear correlation and symmetrical uncertainty as the correlation measure to calculate the similarity for continuous and discrete features respectively. In fact, to average the comparability of the correlation values which come from different measurement systems, we multiply each correlation of sim(X_i,C) (correlation between X_i and class label) by the corresponding averaged correlation of the same types of features and the same approach for reducing redundant. The ECMBF algorithm requires two parameters, the relevance threshold α and the redundancy threshold β . These parameters are employed to distinguish weak irrelevance/relevance and redundancy, respectively. The choice of the two parameters can significantly affect the quality of the selected feature subset which is the crucial problem in this algorithm.

2.2.11 Maximum Relevance–minimum Multi-Collinearity (MRmMC)

Maximum relevance minimum multi-collinearity (MRmMC) is a new relevancy-redundancy approach proposed in Senawi et al. (2017) for feature selection and ranking. MRmMC measures feature relevance by correlation properties based on conditional variance while redundancy elimination is evaluated according to multiple correlation assessments using an orthogonal projection scheme. This method

can efficiently prevent some shortcomings of existing methods such as mRmR inherited from the original MIFS algorithm introduced in Battiti (1994). It is known that MIFS has a drawback in that its performance relies on the choice of the parameter beta for controlling and shrinking the redundancy; the optimal choice of the parameter beta, however, strongly depends on the problem to be solved. MRmMC does not require any pre-specification or determination of thresholds for parameter settings.

Table 2.1
provides a description of filter methods discussed in this thesis.

Filter methods	Uni /Mutivariate	Ranker/Subset	Correlation/instance-based
Information Gain	Univariate	Ranker	Correlation-based
Relief	Mutivariate	Ranker	Instance-based
Fisher score	Univariate	Ranker	Instance-based
Mutual Information	Univariate	Ranker	Correlation-based
mRmR	Mutivariate	Ranker	Correlation-based
Pearson	Univariate	Ranker	Instance-based
CFS	Mutivariate	Subset	Correlation-based
$\mathcal{M}_d$	Mutivariate	Subset	Correlation-based
ECMBF	Mutivariate	Subset	Correlation-based
MRmMC	Mutivariate	Subset	Correlation-based

2.3 Wrapper methods

Instead of ranking each feature individually as in filters, the Wrapper approach evaluates feature subsets using the classifier's prediction performance as a black box (see Figure 2.1). It uses prediction performance as a criterion function to guide the search for the best feature subset and tries to find features that maximize it. This approach takes into account the interactions between features as opposed to Filters which helps hugely to achieve better performance than filters. Moreover, wrappers require a massive amount of computations because many models must be constructed from scratch during the search process. The used search strategy is crucial because with the increase of data dimensionality, using inappropriate search strategy could be NP-hard problem Woeginger (2003) because the search becomes infeasible. Especially, exhaustive search methods become computationally impractical for larger datasets. Consequently, diverse methods and algorithms such as sequential search and evolutionary algorithm are further developed and introduced to achieve good experimental performance with practical computational costs. However, since wrapper methods are relaying on induction algorithm, they are more demanding in terms of computational complexity than Filter and Embedded methods.

Many wrapper methods are proposed in the literature, in this subsection we will focus on the most commonly used methods under wrapper approach and those used through this thesis.

2.3.1 Branch and bound Algorithm

In 1977, Narendra and Fukunaga have proposed new feature subset selection method called " A Branch and Bound algorithm for feature subset selection ". The algorithm is very efficient in selecting feature subsets and avoiding exhaustive search by rejecting sub-optimal subsets without direct evaluation and guarantees that the selected subset yields the globally best value of any criterion that satisfies monotonicity Viswanath et al. (2007). The central idea of BB algorithm is the following: 1) starting from the root of the tree, the successors of the current node are counted in an ordered list. 2) The new node is the successor for which the objective function is maximum. 3) Maintain the best feasible solution obtained so far as a bound then, the algorithm moves to the next higher level. If the evaluation function value is lower than the identified bound, then the search from that node should stopped. Otherwise, additional searching from that node is necessary.

2.3.2 Sequential forward selection

Sequential forward selection (SFS), which is a greedy search algorithm, is one of the most basic used feature selection algorithms. It begins with an empty set and greedily adds one feature at a time until the performance cannot longer be improved. SFS performs well when the number of representative features (optimal subset) is small Nogueira (2018). The drawback of this method is that once a feature is added to the subset, it cannot be removed afterwards, which sometimes leads to sub-optimal results. The following steps give an in-depth description of the SFS method.

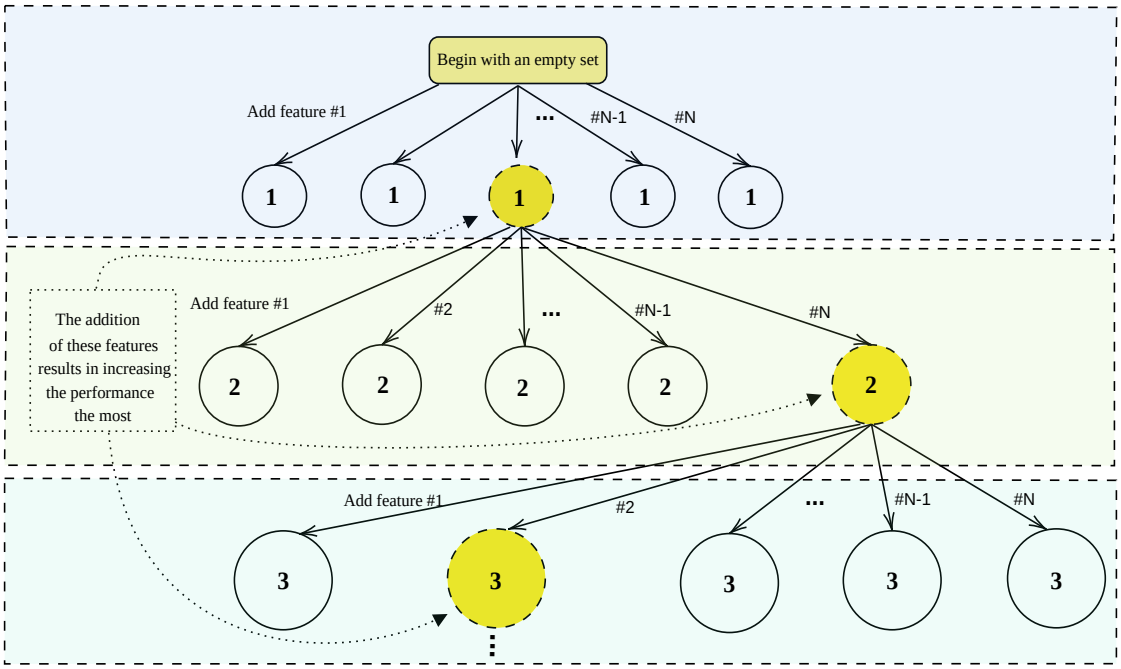


Figure 2.2 – Illustration of three steps of SFS subset selection algorithm. The algorithm starts with an empty set of features and sequentially adds a single feature from available features. After evaluating the classifier performance, resulting by adding each available feature, it finally selects the feature of which addition increases the accuracy the most. This feature is afterwards added to the set and is carried into the next iterations. The process is repeated by attempting to add another feature from the $N - 1$ remaining ones, until the stopping criterion is reached.

2.3.3 Sequential backward elimination

As opposed to forward selection, Sequential backward elimination (SBS) starts with a full set of features and iteratively removes features one at a time. At each iteration, the feature whose removal results in the largest increase (or smallest decrease) in the evaluation function value is removed.

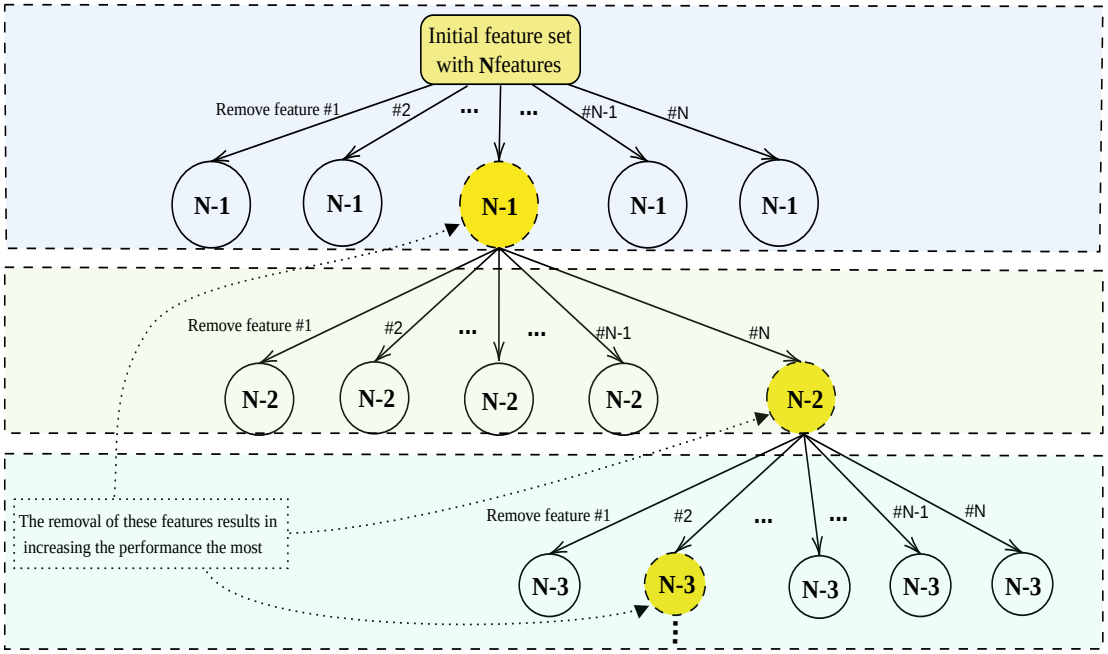


Figure 2.3 – Illustration of three steps of SBS subset selection algorithm. The algorithm starts with all N features and sequentially removes a single feature and evaluates the performance using a wrapped classifier. Feature, which removal increases the accuracy the most is added to the final set and the process is repeated by attempting to remove another feature from the $N - 1$ remaining ones.

2.3.4 Stepwise (Bi-directional search)

Stepwise technique applies forward and backward simultaneously to address the problem where features are added/ removed earlier in feature selection procedure. At each iteration, stepwise method may add and remove features.

2.3.5 Recursive Feature Elimination

The RFE selection method Guyon et al. (2002), which is a backward selection of predictors and greedy optimization algorithm, is a recursive process. It ranks features according to some measures of their importance. It starts with the whole feature space and at each iteration, features' importance are assessed. Then, the less important feature is eliminated. To speed up the elimination process, another alternative is proposed to remove a group of features at once. The reason why the recursion is necessary is the possibility of the substantial importance change of each feature when assessed over a different feature subset the moment of performing the elimination process (most probably for highly correlated features). The order in which features are discarded during the recursion process is used to rank features.

2.3.6 Support Vector Machine Recursive Feature Elimination (SVM-RFE)

In Guyon et al. (2002), the authors introduced a hybrid feature selection method where a support vector machine is based on recursive feature elimination (SVM-RFE). It starts with the entire feature space and iteratively features are discarded one at a time without any backtracking. This greedy top-down strategy computes the normal vector of a decision hyper-plane created by a linear support vector machine (SVM) classifier. Projections of the normal vector to the coordinate system are subsequently used to assess the ranking of individual features. The features with the highest-ranking are kept, and those with the lowest ranking are removed at each iteration. This process could be accelerated by removing more than one feature in each iteration. This procedure is repeated until the desired number of features is attained, or the performance cannot be improved any longer.

2.3.7 Random forest Recursive Feature Elimination (RF-RFE)

RF-RFE is a feature selection where random forest model is used to assess the importance of features. First, a full model is constructed using the entire feature space, then at each stage of the search, the least important features are iteratively eliminated prior to rebuilding the model. Second, the whole process iterated until all features are exhausted. Finally, features are ranked according to their elimination order.

2.4 Embedded methods

In contrast to wrapper strategy that employs a heuristic search guided by the performance of the classifier, embedded strategy provides a trade-off alternative between filter and wrapper methods. It uses the learning process itself to both performs feature selection and builds an optimized classifier. Embedded approach, which is an intermediate solution between filters and wrappers, selects features that are evolved during the learning process relying on the evaluation criterion of the classifier which generally leads to less computational cost than wrappers. The most widely used embedded methods are the regularization techniques.

2.4.1 LASSO (L_1-Regularization)

In 1996, Robert Tibshirani Tibshirani (1996) introduced a powerful regularization feature selection method called the Least Absolute Shrinkage and Selection Operator (LASSO). The LASSO method adds a penalty to the sum of absolute values of the machine learning model parameters. The added constraint shrinks (regularization) some coefficients to zero. During the feature selection procedure, features with non-zero coefficient after shrinkage phase are selected to be part of the model, and those with exactly zero-coefficient are to be discarded. To control the strength of the regularization, a tuning parameter λ (regularization parameter) is used. When λ is sufficiently large, coefficients are forced to be exactly zero which results in reducing dimensionality. Indeed, the larger λ is, the more coefficients are shrinked to zero. On the other hand, if $\lambda = 0$ this means that no regularization is applied (Ordinary Least Square). LASSO has many advantages. It is effective

at reducing variance because the coefficients of the misleading features are penalized and removed, consequently, it helps avoiding over-fitting problem, thus, better generalization ability. Moreover, LASSO is very useful at interpretability enhancement by canceling irrelevant features. There are different mathematical forms to introduce LASSO method; According to Robert Tibshirani, The lasso estimate is defined by the solution to the L_1 optimization problem.

$$Minimize\left(\frac{(\|Y-X\beta\|_2^2)}{n}\right) \text{ subject to } \sum_{j=1}^{k}\|\beta\|_1 < t \tag{2.9}$$

Where t is the upper bound for the sum of the coefficients. This optimization problem is equivalent to the parameter estimation that follows:

$$\hat{\beta}(\lambda) = argmin_\beta(\left(\frac{\|Y-X\beta\|_2^2}{n} + \lambda\|\beta\|_1\right) \tag{2.10}$$

Where $\|Y-X\beta\|_2^2 = \sum_{i=0}^{n}(\|Y_i-(X\beta)_i\|^2)$ and $\lambda \geq 0$ is the regularization parameter that control the amount of shrinkage. The lasso method has some limitations:

- From each group of correlated features, LASSO tends to select one feature from each group and neglects other features.
- In datasets where m (number of instances) is small, and n (number of features) is large, LASSO selects at most m features before it saturates.

2.4.2 RIDGE (L_2-Regularization)

Ridge or L_2-Regularization Naseriparsa et al. (2014)is a regularization technique. It adds a squired magnitude of the model parameters as penalty term to the objective function. As opposed to LASSO, which provides a sparse set of features, Ridge shrinks model coefficients close to zero and not exactly zero. The main difference between LASSO and Ridge techniques is that LASSO shrinks the unimportant and uninformative feature's coefficients to zero, which leads to cancelling useless features altogether. On the other hand, Ridge results in non-zero coefficients, which is more helpful for feature interpretation.

2.4.3 Random forest for feature selection

Random forests (RF) Breiman (1999, 2001) is among the most used machine learning algorithms not just for its excellent prediction accuracy but also for its ability to select informative variables with its associated variable importance measures Gini index and Mean decrease accuracy Menze et al. (2009); Saeys et al. (2008); Genuer et al. (2008).

- Gini index: measures how well a split on each variable is separating the samples of the two classes in this given node averaged over all trees Genuer et al. (2008).

- Mean decrease accuracy: This measure is computed when data are permuted in OOB (out-of-bag) samples: RF importance variable is the difference between the prediction error recorded on OOB samples and the prediction error after permuting the values of data averaged over all trees in the forest Sandri and Zuccolotto (2006); Hapfelmeier and Ulm (2013).

2.4.4 XGBoost based-model for feature selection

XGBoost Chen and Guestrin (2016), a gradient boosting decision tree based on the usage of regularized learning and cache-aware block structure tree learning for ensemble learning. L represents the loss function; f_t represents the t_{th} tree and $\Omega(f_t)$ is regularized term. The second-order Taylor series of L at the t_{th} iteration is:

$$L^{(t)} \simeq \sum_{i=1}^{k} \left[l(y_i,\ y_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x) \right] + \Omega(f_i) \tag{2.11}$$

where g_i and h_i denotes the first and second order gradients. During our training of XGBoost, we uses gain to determine the optimal split node.

$$gain = \frac{1}{2} \left[\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \lambda} \right] - \gamma \tag{2.12}$$

where I_L and I_R represent samples of the left and right nodes after the segmentation, respectively. $I = I_L \cup I_R$. λ and γ are the penalty parameters. Gain represents the gain score for each split of a tree, and the final feature importance score is calculated by the average gain. The average gain is the total gain of all trees divided by the total number of splits for each feature. The higher the feature importance score of XGBoost is, the more important and effective the corresponding feature is (see Figure 2.6 and 2.7).

2.5 Ensemble Feature Selection

Recently, the robustness and stability of feature selection method become an important aspect Li and Yang (2005). Since single feature selector cannot be efficient to ensure unbiased and reliable optimal results (good performance and stability), many algorithms have been introduced in the literature to explore the ability of ensemble technique by combining a collection of weak and unstable feature selectors. Generally, ensemble feature selection techniques yield a more robust and stable feature subsets than a single feature selector. The crucial factor that leads to the success of ensemble methods is the degree of diversity of the models in the ensemble. There are different ways to ensure diversity: 1) training different learning algorithms, 2) training the same learning algorithm with different parameterized versions, 3) Build multiple learning models using different generated training sets. In addition to the idea of using more than one classifier or feature selector to select a stable feature subset, an aggregation technique, which combines and aggregate feature selection methods (see Figure 6.3), is introduced and discussed by Akhiat et al. (2019); Saeys et al. (2008); Opitz (1999). The EFS might

avoid the risk of inconsistent results and tends to produce a better approximation of the optimal subset since individual feature selectors could lead to different suboptimal solutions.

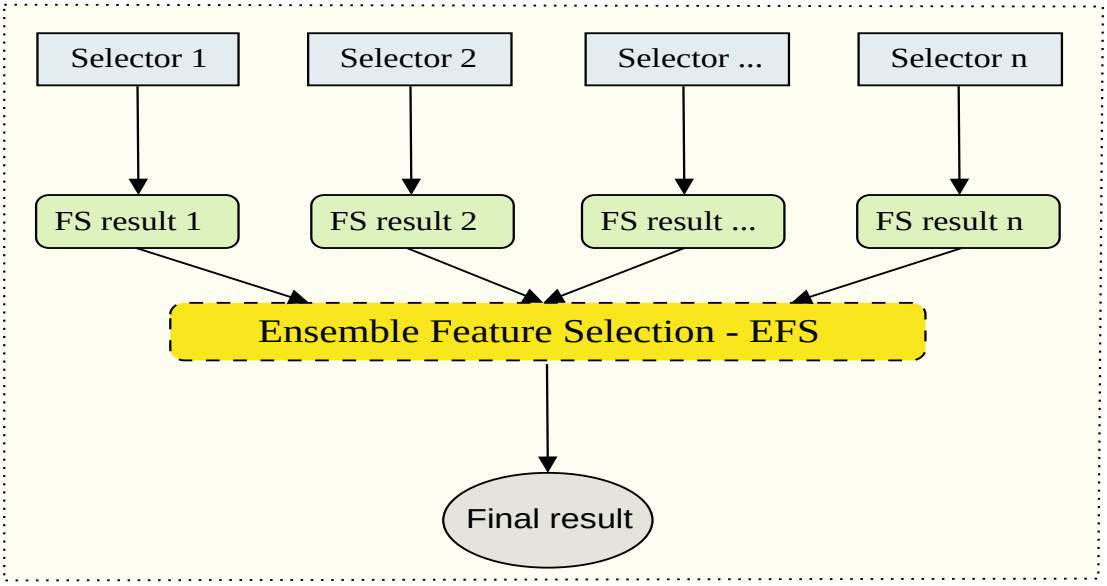


Figure 2.4 – Ensemble Feature selection design where many selector' results are combined to provide a stable ensemble feature selection.

Definition (stability)

The stability of feature selection method can be defined as the variation in the feature selection results due to small changes in instances or attributes level of datasets. There are two steps to create an ensemble feature selection method:

1. The first step consists of running different feature selection methods; each method provides its own feature subset.
2. In the second step, the results of each single feature selector are aggregated which can be done by weighted voting or counting the most frequently selected features.

2.6 Hybrid feature selection

Hybrid and ensemble methods are the latest developments in feature selection field. Rather than using a single feature selection approach, Hybrid method can be formed by combining two different methods from different FS categories (e.g. filter and wrapper), two methods of the same evaluation function, or two feature selection approaches. Hybrid FS methods take the best advantages from both approaches by combining their performance. The most common hybrid method is the combination of filter and wrapper methods Hsu et al. (2011); Sebban and Nock (2002); Wang and Liu (2016).

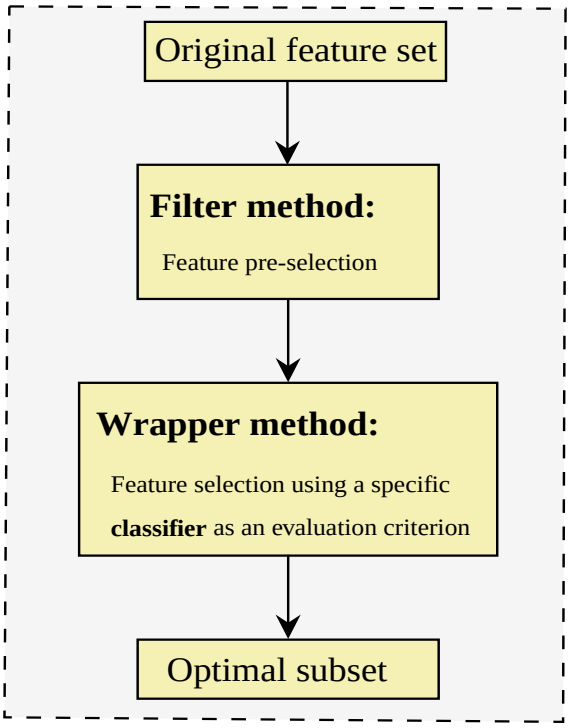


Figure 2.5 – Hybrid feature selection diagram.

2.7 A Comparative Study

In this section, we will provide a critical analysis of existing feature selection approaches, adding two extra FS categories to the traditional FS categorization (Hybrid and Ensemble feature selection) since the majority of existing surveys such as those in Wang and Liu (2016); Guyon and Elisseeff (2003); Alelyani et al. (2013) did not mainly cover these two recent categories. In addition, empirical comparisons are conducted to assess and evaluate the performance of nine FS methods and their applicability as well.

2.7.1 Feature selection evaluation

In this section, various comparisons between feature selection methods of different categories are performed to empirically expose the key difference between each feature selection category and between each method within the same category. We therefore focus on comparing the following nine methods for feature selection which are previously discussed: ***ReliefF*** 2.2.2, ***Chi-square (Chi-2)*** 2.2.4 and ***Mutual Information (MI)*** 2.2.5 as filter methods. The wrapper methods we have used are the following: ***Sequential Forward Selection (SFS)*** 2.3.2, ***Sequential Backward Selection (SBS)*** 2.3.3 and Recursive feature elimination using SVM ***(RFE-SVM)*** 2.3.6, ***LASSO*** 2.4.1, ***RIDGE*** 2.4.2 and ***Random Forest (RF)*** 2.4.3 as embedded methods. In the next subsections, we provide the summarized descriptions of each one of these methods.

Algorithm: The pseudo-code of the evaluation and testing strategy.

```
DS - set of datasets
FS - set of feature selection methods
C - set of classifiers
For i in DS do:
  Train ←80% of i dataset
  Test  ←20% of i dataset
  For j in C do:
    For k in FS do:
      Accuracy ←Performance evaluation of the
                selected subset on  Test set
Result(i,j,k)←Accuracy
```

2.7.2 Feature selection and learning algorithms

In this experimental section, we tested the performance of nine feature selection methods from different FS categories when used in conjunction with three classification algorithms to obtained more reliable results. We use classification accuracy to evaluate the classification performance. Generally, the higher the classification accuracy, the better the selected features are. We implemented our experiments using statistical package scikit-learn Cai et al. (2018). The used implementations of the feature selection methods are the following:

- **ReliefF**: We have used an implementation of the method relief in python in the package ReliefF. The chosen parameters were: number of randomly selected examples was 10% of all examples and the threshold was 0.01.
- **Chi-square** and **MI**: relying on the function SelectKBest of the scikit-learn package of python, we implemented Chi-square and MI method. The score function parameter is set as follows: score-func=chi2 for Chi-square and score-func=mutual-info-classif for MI.
- **Random forest**: Implementation from the ensemble library of scikit-learn, used with default parameters (500 trees in a forest).
- **SFS** and **SBF**: An implementation of the SFS and SBS algorithms was used based on the function SFS from the package mlxtend. The chosen parameters were: forward set to TRUE for SFS and False for SBS. We used the Decision tree as the wrapped learning algorithm for both methods.
- **LASSO** and **RIDGE**: These two methods are implemented using linear-model package. For both methods, the used regularization terms are varying in the interval alpha:[0.1,0.01,1.,1,2] and max-iter=10000.

After selecting the best feature subset using the previously mentioned FS methods for each dataset, the choice for building the classification models is among three different machine learning classifiers. These classifiers are commonly used in data mining. The classification accuracy is used to compare the FS methods in terms of their robustness to classify a given dataset.

- **Decision trees**: implementation from the package sklearn.tree was used, information gain was used as the splitting criterion.

- **Random forest**: implementation from the python package randomForest was used. The number of trees in a forest was set to n-estimators=50 and max-depth=3.
- **Support vector machine (SVM)**: a svm.SVC implementation from the Python package is used. SVM is used with a linear kernel and the cost parameter was set to $C = 1$.

2.7.3 Experimental evaluation and results

The recorded detailed experimental results are presented in table 2.5 for filters, table 2.6 for wrappers and in table 2.7 for embedded methods. In table 2.3, we provide a summary of all detailed results to enhance their clarity and the readability. The obtained results are averaged for each of the tested classifier in the summary row (mean) at the bottom of the table 2.3 (each classifier was tested in 99 experiments = 11 datasets × 9 FS methods, this results in 297 experiments), show that the accuracies of KNN, SVM and RF classifiers were significantly improved after applying feature selection methods in our evaluation procedure despite the considerable reduction (using just the first fifteen best features out of the whole original feature space). The significant increase in classification accuracy was achieved in 118 experiments (40%), in 33% the accuracy were decreased, and in 27% of datasets, the classification accuracy remains stable (in fact, maintaining the performance accuracy of the classifiers despite the dimensionality reduction is very advantageous). This considerable dimensionality reduction enables the construction of fast models (training models with a reduced feature subset is faster than training them using the entire feature space) and decreases the storage and memory requirement.

- By analyzing the performance attainability of filter methods (see table 2.5 and table 2.3), it is revealed that ***MI*** feature selector for KNN classifier is able to increase the classification accuracy in the majority of datasets (9 out of 11) followed by ***Chi-square*** for RF classifier (8 out of 11). As a conclusion, ***MI*** and ***Chi-2*** are the most aggressive in terms of feature removal with accuracy maintenance (without losing accuracy) of the final prediction. This finding is supported also in Yang and Pedersen (1997).
- As it is illustrated through the summarized results of wrappers in table 2.3, it is clear that ***RFE-SVM*** outperforms ***SBS*** and ***SFS*** as it increases the accuracy of the classifiers in 40% of experiments. Moreover, the detailed results of table 2.6 have shown that wrappers could achieve better performance than filters in terms of accuracy as they take into account the underlying interactions between features. However, the individually irrelevant features may become weakly or strongly relevant when interacted and combined with other features.
- Table 2.7 and table 2.3 demonstrate that RF classifier could be applied as a selector depending on its variable importance technique aiming to select the most informative features. The results show that the classification accuracy increased in 18 experiments (55%), the performance deteriorates in 15% (5 out of 33) and remains unchanged in 30% of experiments. These results are

followed by ***LASSO*** selector. For ***RIDGE*** selector, it increases the performance just in 4 experiments out of 33 and the performance of the classifiers deteriorates in the most experiments (55%). The performance degradation of the classifiers after applying ***RIDGE*** as a selector can be explained by the fact that ***RIDGE*** method can only shrink coefficients close to zero and not exactly zero as in ***LASSO*** method. This enable to select the uninformative features especially when we deal with large number of features (for example: madelon, ds1.100 and clean).

Table 2.3
The summarized results for all tested feature selection methods (rows) and classifier algorithms (columns). The results are presented in the following triplet form +/ - /=, where + denotes the number datasets in which the classification accuracy notably increased, - denotes the number datasets in which the classification accuracy significantly decreased and = denotes the datasets in which the classification accuracy remain unchanged.

	FS methods	SVM +/-/=	RF +/-/=	KNN +/-/=	Average +/-/ =
Filters	RelifF	4/3/4	5/4/2	5/4/2	14/11/8
	MI	4/2/5	6/4/1	9/1/1	19/7/7
	Chi-square	3/4/4	8/2/1	5/5/1	16/11/6
Wrappers	SFS	1/5/5	2/4/5	5/3/3	8/12/13
	SBS	2/4/5	3/5/3	6/5/0	11/14/8
	RFE-SVM	3/3/5	3/3/5	7/3/1	13/9/11
Embedded	RF	4/3/4	7/0/4	7/2/2	18/5/10
	LASSO	5/4/2	4/2/5	6/3/2	15/9/9
	RIDGE	1/5/5	0/7/4	3/6/2	4/18/11
	Average	27/33/39	38/31/30	53/32/14	118/96/83

Besides classification accuracy, execution time is another critical property of feature selection methods especially for large datasets. Table 2.4 lists the running time (in seconds) of the implemented FS methods. It is clear that Filter methods used much less time on average in comparison to the other approaches (Wrapper and embedded). As wrappers rely on learning algorithms to evaluate each generated subset, they are computationally costly compared to the filter approach. This is even clearly demonstrated when the size of the dataset becomes larger (as shown for the datasets ds1.100, clean and madelon). Embedded methods could be a good alternative especially when the feature space is of high dimension.

Table 2.2
Advantages and disadvantages of Feature selection approaches.

	Strengths	limits
Filters	- Independent of learning algorithm. - Fast execution. - Suitable for large-scale data. - Good generalization ability.	- The correlation between features is neglected. - Allow redundant features to be selected. - No interaction with the classifier.
Wrapper	- More accurate than filters. - Take into account interactions between features. - Identify feature dependencies	- More computations requirement. - Over-fitting problem can be a serious problem. - Building a model from scratch at each examined subset. - Some features may be eliminated earlier, while they should not.
Embedded	- Faster than wrappers. - Accurate. - Take into account interactions between features. - Identify feature dependencies.	- Specific to learning algorithm. - Classifier-dependent selection.
Hybrid	- Higher performance than filters. - Less computational complexity than wrappers. - Less prone to over-fitting problem.	- Specific to learning algorithm.
Ensemble	- More stable than other categories. - More flexible upon high dimensional data.	- Hard to interpret and to understand.

Table 2.4
Running time of feature selection methods (in seconds). m denoted the number of examples and n denoted the number of features in each dataset.

	Filter			Wrapper			Embedded		
	ReliefF	MI	Chi-2	SFS	SBS	RFE-SVM	LASSO	RIDGE	RF
Spambase(m=4601,n=57)	2.80	6.90	1.90	13.18	79.26	41.02	4.48	6.53	2.43
Chess(m=3196,n=36)	1.92	4.06	2.12	5.37	8.88	4.81	3.26	5.19	1.02
Clean(m=6598,n=167)	5.21	21.48	5.61	111.76	4483.17	117.15	3.07	4.43	5.71
eightr(m=200000,n=306)	0.93	2.50	0.90	11.94	135.25	3.39	1.37	2.23	1.51
caravan(m=5823,n=86)	1.44	4.99	0.98	16.67	394.82	28.64	3.53	5.78	2.44
ionosphere(m=351,n=34)	1.35	1.70	1.42	4.72	6.66	1.50	1.71	3.52	1.40
Credit card(m=30000,n=24)	45.54	25.00	46.32	45.05	78.53	100.85	42.44	55.71	13.42
ds1.100(m=26733,n=100)	104.85	54.82	115.20	237.54	>2.12 hours	102.09	11.07	17.42	20.91
sonar(m=208,n=61)	1.07	1.79	1.43	5.86	19.86	1.54	1.52	2.99	0.77
madelon(m=4400,n=500)	160.19	210.2	164.3	108.59	>70 hours	>4.5 hours	572.87	576.79	188.23
hepatitis(m=155,n=19)	1.28	1.48	1.31	3.17	2.33	1.48	1.54	2.96	141.20

To conclude, our empirical evaluation reveals that filter methods may be very efficient in dealing with high-dimensional data as they neglect the underlying interactions among features. Moreover, when the interpretability problem is the concern, filters can be a good alternative. In contrast to filters, wrappers may produce better performance, but since they are specific for the supervised learning model, generally, they suffer from the over-fitting problem. In the wrapper approach, for each examined subset, models should be constructed from scratch, which makes them very expensive in terms of execution time. Embedded methods, which use a part of the learning process of the model for feature selection, reduce the computational cost of wrappers by preventing the reclassification of different feature subsets as performed in wrappers.

Table 2.5
The detailed results for each testing dataset, FS filter method and classification algorithm. The performance attainability of each filter method is recorded using three classification algorithm (SVM, RF and KNN). For each testing dataset, the initial classification accuracy of the baseline models (without selection) is recorded as well.

	Without Feature selection			With feature selection (Filter approach)								
				ReliefF			MI			Chi-2		
	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN
Spambase	0.65 ±0.02	0.89 ±0.04	0.74±0.11	0.58±0.0	0.88±0.04	0.82±0.03	0.58±0.00	0.88±0.11	0.85±0.04	0.64±0.00	**0.89±0.03**	0.70±0.05
Chess	0.90∓0.05	0.92±0.04	0.89∓0.03	0.79±0.06	0.79±0.06	0.77±0.06	0.88±0.04	0.91±0.02	0.90±0.03	0.88±0.04	0.91±0.02	**0.93∓0.05**
Clean	0.92∓0.02	0.91±0.02	0.94±0.01	0.90±0.00	0.91±0.01	0.92±0.02	0.93±0.00	0.93±0.01	**0.94±0.00**	0.89±0.01	0.92±0.01	0.92±0.02
eightr	0.93∓0.01	0.93∓0.01	0.93±0.02	0.93±0.01	0.93∓0.01	0.92±0.01	0.93±0.01	0.93±0.02	**0.95±0.02**	0.93±0.01	0.93±0.01	0.93∓0.03
caravan	0.93±0.02	0.93∓0.03	0.92±0.01	**0.93∓0.00**	**0.93∓0.00**	**0.93±0.00**	**0.93±0.00**	**0.93±0.00**	0.93±0.01	**0.93±0.00**	**0.93±0.00**	0.92∓0.02
ionosphere	0.62±0.07	0.67∓0.09	0.65±0.30	0.62±0.07	0.53±0.22	0.62∓0.23	0.62±0.07	0.74±0.18	**0.79±0.16**	0.62±0.07	0.71±0.20	0.72±0.31
Credit card	0.78±0.00	0.80±0.01	0.76±0.01	0.78±0.00	**0.80±0.01**	0.76∓0.01	0.78±0.00	**0.80±0.01**	0.75±0.01	0.77±0.01	0.79±0.01	0.75±0.01
ds1.100	0.97±0.00	0.97±0.01	0.96±0.02	**0.97±0.00**	**0.97±0.00**	**0.97±0.00**	**0.97∓0.00**	**0.97±0.00**	0.97±0.02	0.96±0.00	0.97±0.00	0.97±0.03
sonar	0.57±0.35	0.61±0.23	0.61±0.35	0.51±0.29	0.67±0.37	0.63±0.28	0.63±0.34	**0.76±0.17**	0.64±0.33	0.64±0.37	0.71±0.31	0.67±0.37
madelon	0.55±0.04	0.61±0.06	0.65±0.07	0.60±0.06	0.61±0.01	0.69±0.08	0.61±0.08	0.66±0.07	**0.70±0.10**	0.59±0.04	0.62±0.08	0.66±0.04
hepatitis	0.81±0.03	0.89±0.19	0.93±0.18	0.81±0.02	0.77±0.27	0.85±0.16	0.81±0.03	0.85±0.26	0.93±0.18	0.81±0.03	**0.93±0.16**	0.93±0.18

Table 2.6
The detailed results for each testing dataset, FS wrapper methods and classification algorithm. The performance attainability of each wrapper methods is recorded using three classification algorithms (SVM, RF and KNN). For each testing dataset, the initial classification accuracy of the baseline models (without selection) is recorded as well.

	Without Featureselection			With feature selection (Wrapper approach)								
				SFS			SBS			RFE-SVM		
	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN
Spambase	0.65 ±0.02	0.89 ∓0.04	0.74∓0.11	0.58∓0.0	0.90∓0.03	0.86∓0.03	0.58∓0.00	0.90∓0.03	0.86∓0.03	0.65∓0.01	0.90∓0.03	0.76∓0.06
Chess	0.90∓0.05	0.92∓0.04	0.89∓0.03	0.87∓0.04	0.91∓0.01	0.87∓0.04	0.87∓0.04	0.91∓0.01	0.87∓0.04	0.89∓0.06	0.91∓0.06	**0.93∓0.04**
Clean	0.92∓0.02	0.91∓0.02	0.94∓0.01	0.84∓0.00	0.92∓0.03	0.95∓0.02	0.84∓0.00	0.90∓0.02	**0.92∓0.03**	0.85∓0.01	0.91∓0.02	0.93∓0.02
eightr	0.93∓0.01	0.93∓0.01	0.93∓0.02	0.93∓0.01	0.93∓0.01	0.92∓0.01	0.93∓0.01	0.93∓0.02	**0.93∓0.01**	0.93∓0.01	0.93∓0.01	0.93∓0.03
caravan	0.93∓0.02	0.93∓0.03	0.92∓0.01	**0.93∓0.00**	**0.93∓0.00**	**0.93∓0.00**	**0.93∓0.00**	**0.93∓0.00**	0.93∓0.01	**0.93∓0.00**	**0.93∓0.00**	0.92∓0.01
ionosphere	0.62∓0.07	0.67∓0.09	0.65∓0.30	0.81∓0.14	0.89∓0.07	0.79∓0.10	0.83∓0.15	0.90∓0.04	**0.80∓0.08**	0.87∓0.10	0.88∓0.08	0.83∓0.18
Credit card	0.78∓0.00	0.80∓0.01	0.76∓0.01	0.78∓0.00	**0.80∓0.01**	0.76∓0.01	0.78∓0.00	**0.80∓0.01**	0.75∓0.02	0.78∓0.00	0.80∓0.01	0.75∓0.01
ds1.100	0.97∓0.00	0.97∓0.01	0.96∓0.02	**0.97∓0.00**	**0.97∓0.00**	**0.98∓0.01**	**0.97∓0.00**	**0.97∓0.00**	0.97∓0.02	0.96∓0.00	0.97∓0.00	0.97∓0.02
sonar	0.57∓0.35	0.61∓0.23	0.61∓0.35	0.53∓0.07	0.60∓0.32	0.66∓0.17	0.61∓0.19	**0.60∓0.23**	0.60∓0.19	0.64∓0.35	0.69∓0.29	0.71∓0.40
madelon	0.55∓0.04	0.61∓0.06	0.65∓0.07	0.71∓0.03	0.74∓0.04	0.80∓0.02	0.77∓0.01	0.80∓0.04	**0.87∓0.03**	0.88∓0.06	0.90∓0.09	0.91∓0.08
hepatitis	0.81∓0.03	0.89∓0.19	0.93∓0.18	0.81∓0.03	0.81∓0.25	0.85∓0.16	0.81∓0.03	0.81∓0.33	0.85∓0.16	0.81∓0.03	**0.87∓0.27**	0.85∓0.16

Table 2.7
The detailed results for each testing dataset, FS embedded methods and classification algorithm. The performance attainability of each wrapper methods is recorded using three classification algorithms ***(REF-SVM, LASSO and RIDGE)***. For each testing dataset, the initial classification accuracy of the baseline models (without selection) is recorded

	Without Feature selection			With feature selection (Embedded approach)								
				RF			LASSO			RIDGE		
	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN	SVM	RF	KNN
Spambase	0.65 ±0.02	0.89 ±0.04	0.74±0.11	0.65±0.01	**0.90±0.04**	0.74±0.08	0.61±0.00	0.64±0.02	0.74±0.03	0.64±0.04	0.86±0.02	0.66±0.04
Chess	0.90∓0.05	0.92±0.04	0.89∓0.03	0.90±0.03	**0.92±0.01**	0.91±0.03	0.78±0.02	0.76±0.03	0.80±0.02	0.73±0.06	0.70±0.05	0.73∓0.04
Clean	0.92∓0.02	0.91±0.02	**0.94±0.01**	0.84±0.00	0.92±0.02	0.93±0.02	0.85±0.00	0.88±0.01	**0.94±0.01**	0.84±0.00	0.88±0.01	0.90±0.02
eightr	0.93∓0.01	0.93∓0.01	0.93±0.02	0.93±0.01	0.93∓0.00	0.93±0.02	**0.94±0.00**	**0.94±0.00**	**0.94±0.01**	0.93±0.01	0.93±0.01	0.93∓0.03
caravan	0.93±0.00	0.93∓0.00	0.92±0.01	0.93∓0.00	0.93∓0.00	0.93±0.01	**0.94±0.00**	**0.94±0.00**	0.94±0.01	0.93±0.00	0.93±0.00	0.92∓0.01
ionosphere	0.89±0.18	0.90∓0.07	0.76±0.07	0.84±0.09	0.90±0.04	0.84∓0.12	0.88±0.06	0.91±0.09	**0.84±0.10**	0.85±0.16	0.85±0.16	0.79±0.12
Credit card	0.78±0.00	0.80±0.01	0.76±0.01	0.78±0.00	**0.80±0.01**	0.76∓0.01	0.78±0.00	**0.80±0.01**	0.76±0.01	0.78±0.00	0.80±0.00	0.74±0.01
ds1.100	0.97±0.00	0.97±0.00	0.96±0.02	**0.97±0.00**	**0.97±0.00**	**0.97±0.02**	**0.97∓0.00**	**0.97±0.00**	0.96±0.03	0.97±0.00	0.97±0.00	0.97±0.02
sonar	0.57±0.35	0.61±0.23	0.61±0.35	0.54±0.28	0.63±0.25	0.73±0.38	0.63±0.13	**0.58±0.38**	0.49±0.28	0.63±0.33	0.61±0.28	0.61±0.29
madelon	0.55±0.04	0.61±0.06	0.65±0.07	0.62±0.08	0.70±0.07	0.82±0.08	0.61±0.04	0.64±0.05	**0.75±0.04**	0.53±0.06	0.53±0.08	0.54±0.11
hepatitis	0.81±0.03	0.89±0.19	0.93±0.18	0.97±0.00	0.97±0.00	0.96±0.02	0.82±0.04	0.85±0.14	0.82±0.04	0.81±0.03	**0.81±0.03**	0.93±0.18

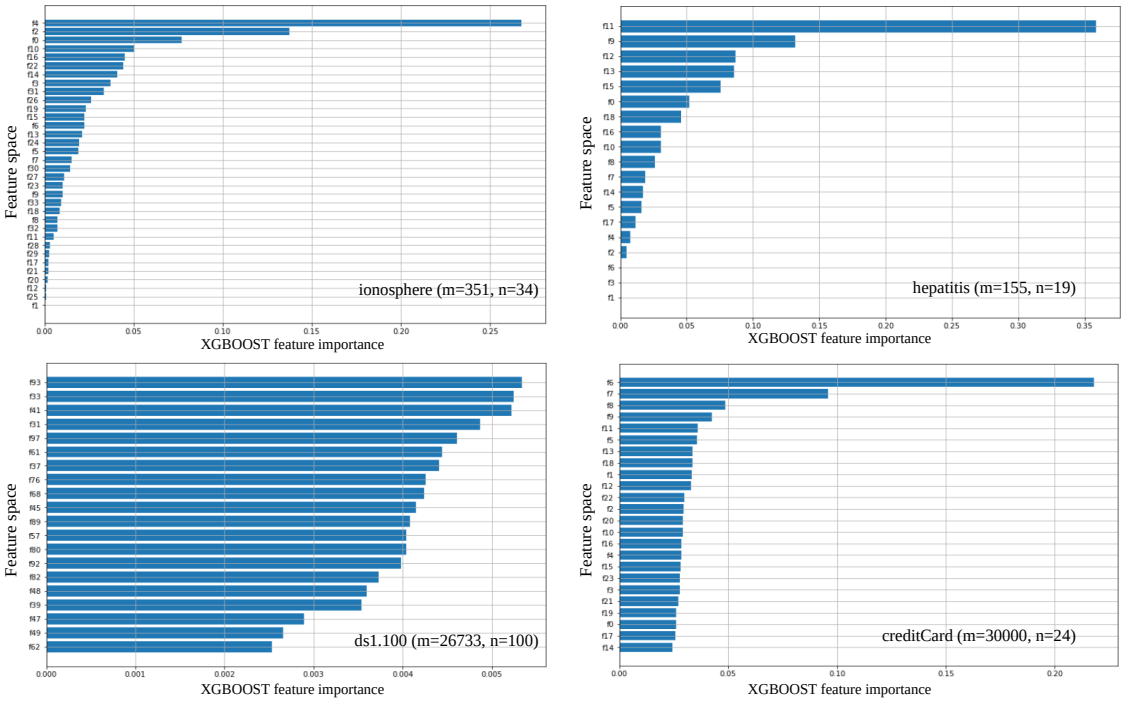


Figure 2.6 – XGBoost feature selection importance on four datasets (ionosphere, hepatitis, ds1.100 and credit card). The x-axis is the XGboost feature importance. The y-axis represents the first twenty heights ranked features.

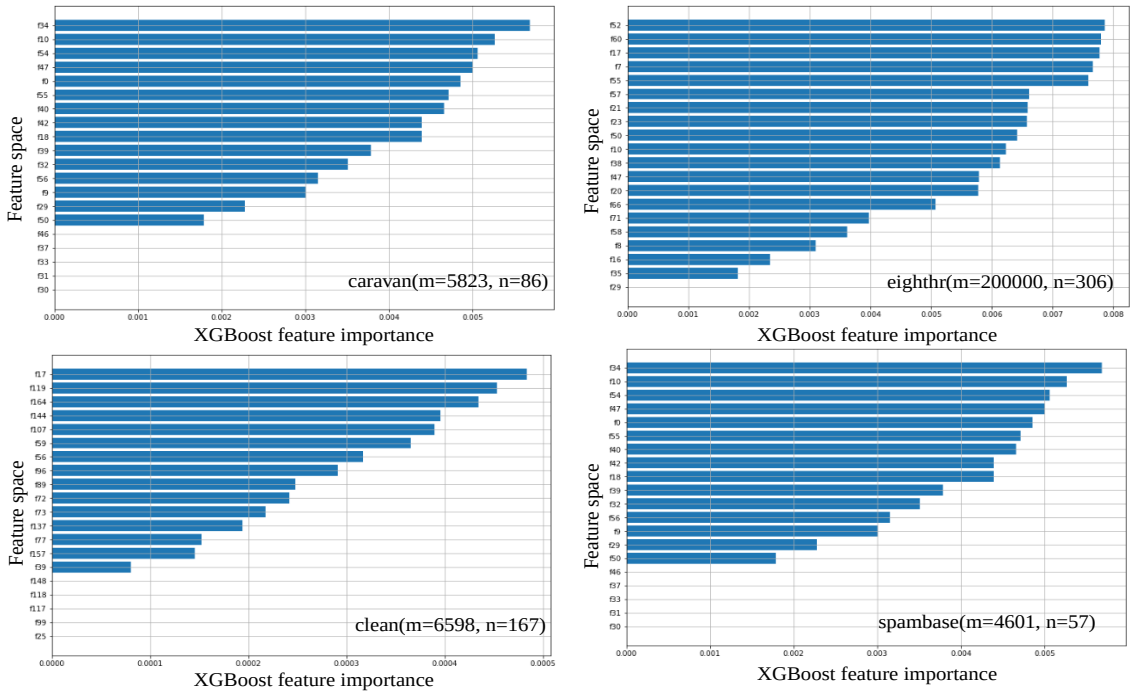


Figure 2.7 – XGBoost feature selection importance on four datasets (caravan, eighthr, clean and spambase). The x-axis is the XGboost feature importance. The y-axis represents the first twenty heights ranked features.

2.8 Guidelines for applying feature selection methods

The naive method for applying feature selection by novice practitioners is to choose one particular feature selection method and apply it on the entire dataset. Then, the best subset is returned and reported as well as the model accuracy if available (in the case of the wrapper and embedded methods). This method is not the best approach as it leads to an exaggerated accuracy and a suboptimal biased and unreliable subset of features, especially; datasets with few instances or with a high number of features (high dimensional datasets). The experts use another advanced strategy of using K-fold cross-validation instead of applying feature selection methods on the whole dataset. First, the entire dataset is divided into k different subsets. Then, $k-1$ subsets (training set) are used to perform feature selection, and the remaining subset (test set) is used to check the validity and the performance of the selected features in the training phase. Second, the whole process is repeated K times on different testing sets, and the average accuracies are reported. This strategy provides accurate estimation results than the first one. One big problem using cross-validation is that the recorded features are not always the same at each iteration. An alternative to deal with this problem and produce robust and accurate estimation is to use an ensemble of models with different features for prediction. Regardless of the provided evaluation technique of cross-validation, it may still lead to biased model performance estimation. Moreover, an unbiased evaluation technique is needed. Forman et al. (2003), the authors have proposed a new evaluation technique. First, the dataset is divided into l different subsets then the feature selection process is performed on $l-1$ subsets. The remaining subset is used to built and evaluate the predictor. The procedure is repeated l times in different testing sets. The proposed technique can be unbiasedly used to compare feature selection methods and to assess their accuracies reliably. In this thesis, we proposed a new evaluation technique, which is a hybridization of both cross-validation and holdout technique. The technique seeks to partition whatever dataset into two complementary sets. The first partition is used to assess the selected features relaying on the cross-validation strategy, and the second part is left at the end to evaluate the performance of different feature selection methods on it. For researchers who want to introduce and develop new feature selection methods, it may be a good alternative for a fair comparison since it ensures reliable and unbiased estimations. For practitioners who want to select the relevant features and remove the irrelevant and redundant ones, it is necessarily practicable to invoke some worth-researching questions as a point of departure to fulfill the feature selection process.

1. The first question: Which feature selection method should be used among those previously discussed in section 1.2.5?
   - To answer the first question, we should first analyze the characteristics of the data.
   - We have to determine the objective of feature selection.
2. The second question: how to evaluate the quality of the selected features?

- Do we have to use cross-validation, dual-loop cross-validation or another technique?
- Do we need to assess features individually or the interactions between them should be considered?

3. The third question: do we need a stable estimation?
   - Ensemble methods could be a good alternative.
   - Sub-sampling data and repeating the analysis for different bootstraps.
4. The fourth question: Which is our main concern: interpretability, high performance of the model or both?

2.9 Summary of the chapter

In this chapter, we have presented the feature selection state of the art methods and their categorization as well. We have started by describing the main difference between feature selection and extraction. Then, we have cited and discussed several fundamental algorithms of each category. Moreover, critical analysis and comparisons are carried out to expose the advantages and drawbacks of each approach and when to use them. In addition,Through this chapter, we have reviewed and empirically evaluated nine feature selection methods from different categories including: ***ReleifF, MI*** and ***Chi-2*** as filters, ***SFS, SBS*** and ***RFE-SVM*** as wrappers, ***LASSO, RIDGE*** and ***RF selector*** as embedded. Each method is evaluated in combination with three well-known classifiers to reliably test their ability to select the best subset. The conducted empirical comparison is carried out using eleven benchmarking datasets to demonstrate the applicability of feature selection techniques. The obtained results show that too much information does not always help machine learning classifiers since it usually implies that a certain amount of features are redundant or irrelevant, and their presence badly affects the performance of the learning algorithms. Therefore, feature selection pre-processing is a mandatory in order to reduce the data dimensionality, provide insight into the data and enhance generalization.

Along with the empirical study, we have overviewed the whole feature selection framework providing to readers and researchers with a holistic overview and preliminary launching groundwork to the feature selection field. For the stability of feature selection method is an overlooked problem, we have spotlighted two recent categories (ensemble and hybrid) since they can produce a consistent and a stable feature subset. Consequently, they could be good alternatives in many machine learning applications.

The follwoing **chapter 3** will be devoted to the designed experimental framework and how to assess the discriminatory ability of the selected features, compare different feature selection methods, and eventually compare the performance of learning algorithms designed for the selected features. Moreover, it will describe the benchmarking well-know datasets employed throught this thesis work to empirically assess and evaluate the proposed feature selection methods

3

Feature selection evaluation methodology

“

“You can have data without information, but you cannot have information without data”.

Daniel keys Maron.

Chapter 3

Feature selection evaluation methodology

Contents

While the second **chapter 2** reviews the most popular feature selection methods in the literature by evaluating their performance as well as summarizing their prominent drawbacks and advantages, this chapter first, presents the feature selection evaluation methodology. In addition, it describes the synthetic and real-world benchmarking datasets employed to evaluate the usefulness of features. Moreover, it discusses the main validation and evaluation technique used in feature selection.

3.1 Introduction

Evaluating and assessing the goodness of feature selection methods is a very important phase. Various evaluation techniques have been proposed in the FS literature Xu and Goodacre (2018); Jain and Zongker (1997b). The common used

evaluation technique is cross-validation and performance measurement based on confusion matrix. In this chapter, we will introduce and discuss the methodology commonly applied to the evaluation of feature selection algorithms. According to the available literature Salzberg (1997); Jain and Zongker (1997b); Kohavi and John (1997); Kudo and Sklansky (2000); Sima et al. (2005), a unique and universal methodology for evaluating feature selection algorithms probably does not exist. The quality of the selected features is usually assessed through the performance attainability of a pre-selected learning algorithm. This raises the problem of performance estimation (error estimation) discussed in Krızek (2008). The question, how to make the best use of the provided data to achieve good performance estimations remains open. Note that most of publicly available benchmark repositories provide data as a one set of labeled examples. Which part of the data should be used for feature selection and which part for the subsequent performance estimation is usually not specified.

Building on the fact that the first phase is devoted to the testification of the performance attainability, the effectiveness and the limits of a feature selection method should be based on synthetic datasets (artificial datasets) Kittler (1978); Pudil et al. (1995); Weston et al. (2000); Gilad-Bachrach et al. (2004). If there is a solid-based preliminary knowledge regarding the optimal features and the valid ability to adjust the experimental conditions, the drawing of more useful conclusions can be successfully achieved. we have evaluated our proposals on these artificial datasets first as conducted in chapter 6. In addition to the employed synthetic datasets, we have evaluated our proposals on several benchmarking real-world datasets (see table 3.1) aiming to scrutinize the ability of each algorithm in selecting the proper feature subset even in extreme cases.

3.2 Evaluation methodology

Before proceeding any further with the proposed feature selection methods, first, we will discuss the designed experimental framework and discover the know-how ways of assessing the discriminatory ability of the selected features, compare different feature selection methods, and eventually compare the performance of learning algorithms designed for the selected features.

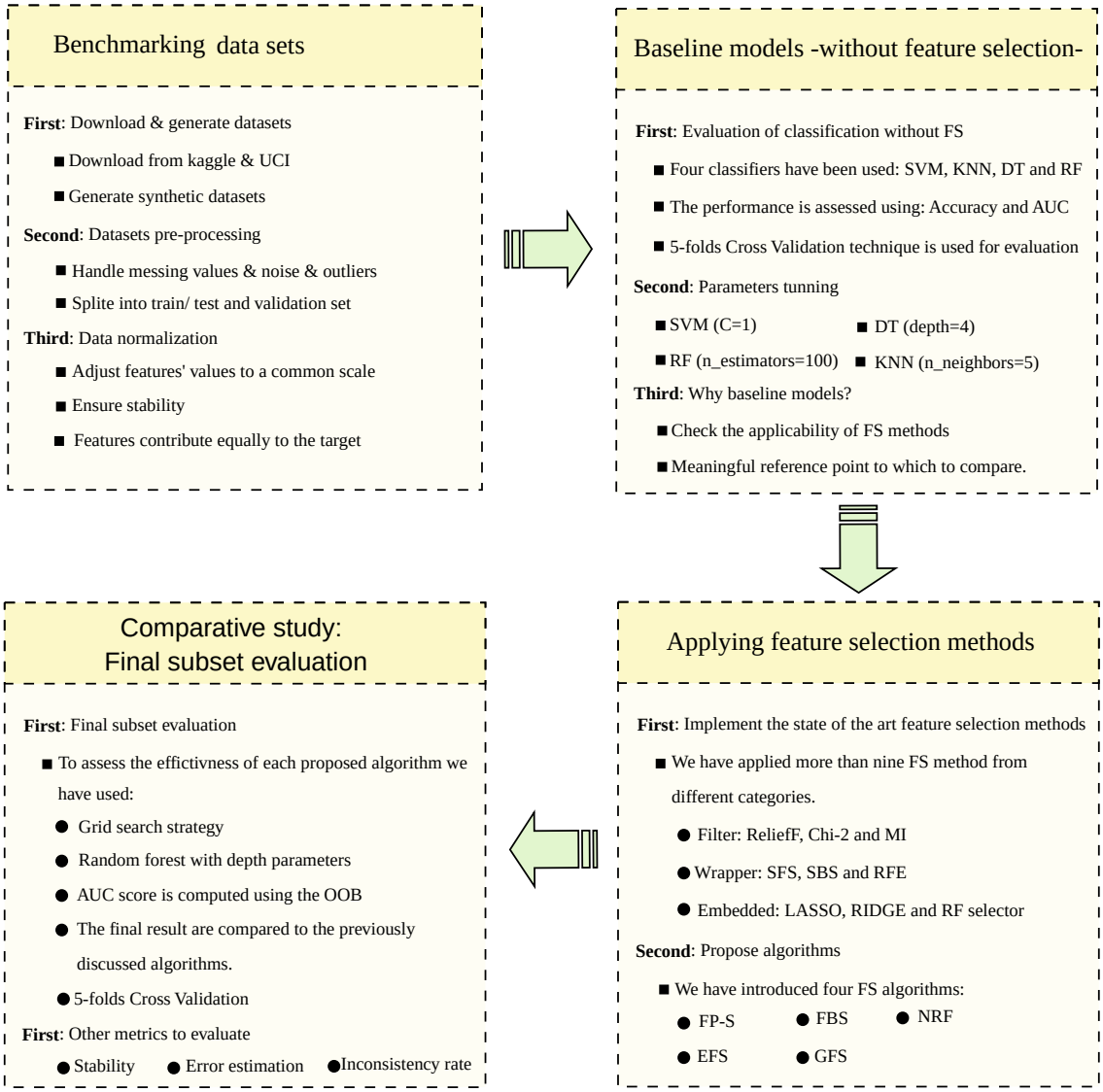


Figure 3.1 – Flow-chart of the design experiment methodology

The Figure 3.1 shows the proposed evaluation methodology. The process starts with data gathering. Then, a normalization step is performed to normalize features of each dataset. Different classification algorithms are trained using the whole features with and without the selected subset generated after applying feature selection methods to assess the classification performance. Finally, a double-headed comparative study of different proposed feature selection algorithms is conducted against both, the state of the art feature selection and models' output without selection of features.

3.2.1 Data collection

In this thesis, fifteen binary classification datasets have been employed in different experimental design aiming to evaluate and check the performance of the proposed feature selection methods. The datasets are chosen to be different in terms of class distribution (balanced or imbalanced), linearity, dataset shift, number of

Table 3.1
Characteristics of the benchmarking datasets.

	#No	Datasets	#Features	#examples	Distribution	class
Standard Datasets	1:	ds1.100	100	26733	3% + / 97% -	2
	2:	Credit card	24	30000	22% + / 78% -	2
	3:	Chess	36	3196	52% + 48% -	2
	4:	Spambase	57	4601	39% + / 61% -	2
	5:	Clean	167	6598	15% + / 85% -	2
	6:	Caravan	86	5823	6% + / 94% -	2
	7:	Madelon	500	4400	50% +/50% -	2
	8:	Santander	371	20000	4% + /96%-	2
	9:	Sonar	61	208	47% + / 53% -	2
	10:	Ionosphere	34	351	64% + / 36% -	2
	11:	Hepatities	19	155	64% + / 36% -	2
	12:	SPECT	32	80	64% + / 36% -	2
	13:	Eye	15	14980	45% + / 55% -	2
	14:	Neumerai	22	96320	50% + / 50% -	2
	15:	eigthr	306	200000	4% + / 96% -	2
High dimensional Datasets	16:	Colon	2000	62	65% + / 35% -	

instances and variables. The datasets, which are publicly available, are collected and downloaded from UCI repository and kaggle platform. An overview of the main characteristics of each dataset is well-presented in table 3.1 followed by a short description about some particular datasets.

Madelon

Madelon dataset, which was part of the NIPS 2003 feature selection challenge, is an artificial dataset. It contains instances grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled +1 or -1. Madelon dataset is a hard problem since it is multivariate and highly non-linear datasets.

Spambase

This is a SPAM e-mail database. The problem is to classify an e-mail message into spam or non spam. There are 57 continuous features denoting word and character frequencies. 39% are spam emails and 61% are not spam.

Sonar

Sonar dataset contains signals generated from a variety of different aspect angles, spanning 90 degrees for the cylinder and 180 degrees for the rock. The task is to distinguish between sonar signals bounced off by metal cylinder and sonar bounced off by cylinder rock.

Ionosphere

This dataset contains 34 autocorrelation functions of radar measurements. The classes were free electrons in the ionosphere. There are 36% of bad returns indicating that a signal passes through the ionosphere and 225 (64%) good returns. The task is to classify signals into passage or obstruction in the ionosphere. Ionosphere dataset used for example in Mitra et al. (2002); Hanzo et al.; Neumann et al. (2004); Skurichina and Duin (2005); Kuncheva (2007); Grabczewski and Jankowski (2005).

Caravan

The data contains 5822 real customer records and 86 variables. The dataset caravan was supplied by the Dutch data mining company sentient data research. The socio-demographic data is generated by zip codes. The task is to discriminate between customers who have insurance policy and those who have not.

Credit card clients

Credit card clients' dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005. This dataset contains 25 variables and 30000 data examples.

Colon high dimensional dataset

The dataset colon contains 40 colon tumor samples and 22 normal colon tissue samples. Each colon sample is characterized by 2000 intensities of 2,000 genes.

3.2.1.1 Data normalization

Datasets usually contains features and variables from different scales and physical units which may affect badly the performance of some machine learning algorithms. Distance based methods (SVM, KNN) and Gradient descent based algorithms (Linear Regression, Logistic regression, Neural Network) are the most sensitive algorithms to feature scaling while others are practically invariant to it Chen and Pau (1995). Normalizing data means adjusting the variable values to a common scale. Further, the propulsive impulse or the incentive that motivates us to normalize data before training any algorithm is the guaranty of a good numerical stability of computations and the making of all the features contribute equally to the concept target. There are different types of data normalization. The most popular normalization technique in the literature is the scaling of the original features so that their instances can have zero mean and unitary variance (means=0 and variance=1).

Assuming that X is the dataset of size m and n features. Let $X = x_{ij}$ where $i = 1, \ldots\ldots, m$, and $j = 1, \ldots\ldots, n$.

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_{ij} \tag{3.1}$$

And

$$\sigma_j^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_{ij} - \mu_j)^2 \tag{3.2}$$

are row means and variances of the input pattern matrix $X = (x_{ji})$, respectively. Here $i = 1, ..., m$ are sample indices and $j = 1, ..., n$ are feature indices. The data transformation can be constructed by:

$$\frac{x_{ij} - \mu_j}{\sigma_j} \rightarrow x_{ij} \tag{3.3}$$

3.2.2 Classification algorithms

Through this thesis work, we have evaluated the performance of several feature selection algorithms when used in conjunction with four classification algorithms (Decision Tree, Random Forest, K-Nearest Neighbors and Support Vector Machine). In this section, we will provide a brief overview of each classifier.

3.2.2.1 Decision Tree

The decision tree DT is a powerful tool for classification and prediction by finding out the patterns or decision rules between data. It is one of the most frequently used data mining methods. DT is made up in the form of a tree and a full tree is built by making child-nodes until each branch reaches the terminal node. The following steps give an in-depth explanation of the DT procedure.

1. Step 1: Determine the Root of the Tree.
2. Step 2: Calculate Entropy for The Classes.
3. Step 3: Calculate Entropy after Split for Each feature.
4. Step 4: Calculate Information Gain for each splitting feature.
5. Step 5: Perform the Split.
6. Step 6: Perform Further Splits.
7. Step 7: Stop the splitting process and return the final Tree.

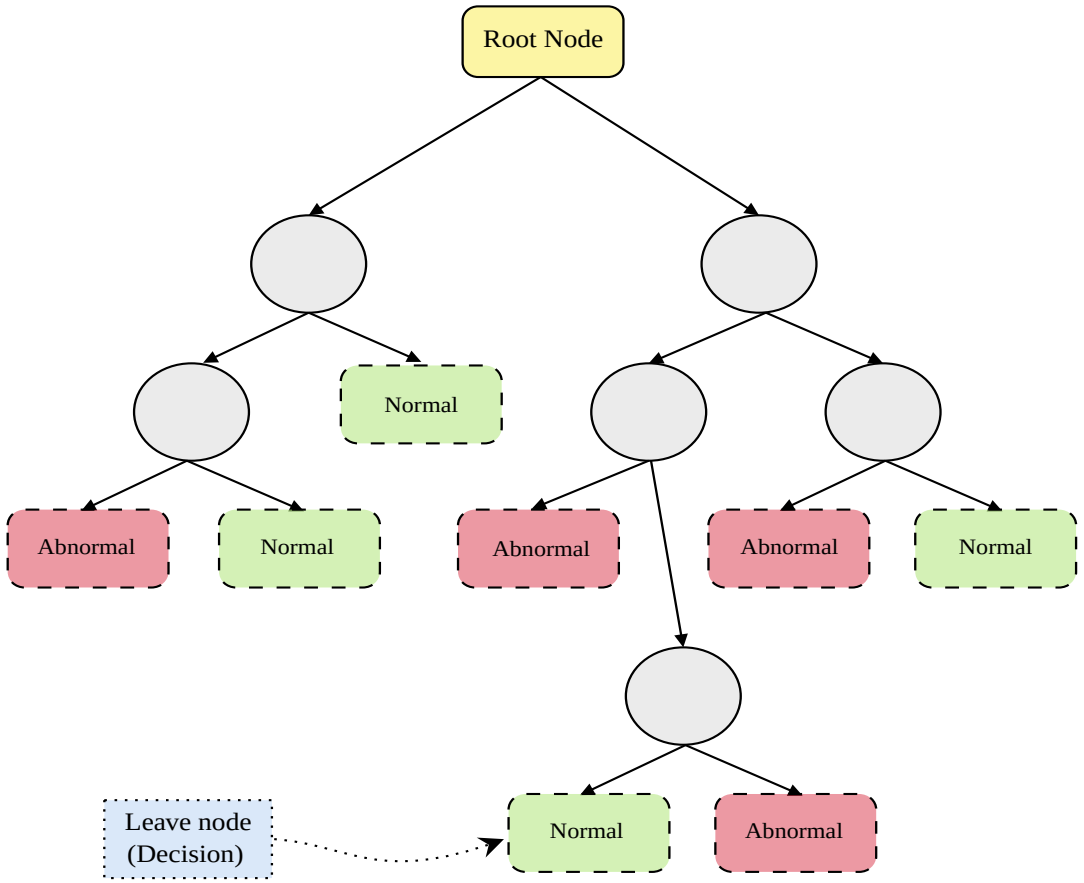


Figure 3.2 – Decision Tree classifier (DT)

3.2.2.2 Random Forest

Random forest is an ensemble method that consists of building an ensemble of decision trees grown from a randomized variant of the tree induction algorithm. In simple words, Random forest builds multiple decision trees (called the forest) and combines them together to obtain a more accurate and stable prediction. The forest it builds a collection of Decision Trees trained with the bagging method Breiman (1996). The algorithm can be explained by the following steps:

1. A bootstrap sample, representing approximately one third of the reference dataset is selected with replacement from the reference dataset.
2. A fully grown tree is generated for each bootstrap sample using a random subset of predictors, which reduces the possibility of over-fitting the tree. Individual trees are fully grown and not pruned to ensure that bias is minimized and the individual error is reduced.
3. Each tree of the RF consists of decision nodes that are produced by randomly selecting a pre-defined number of covariates and the best split is chosen from among these variables.
4. New data are predicted by majority voting of all predictions in the forest.

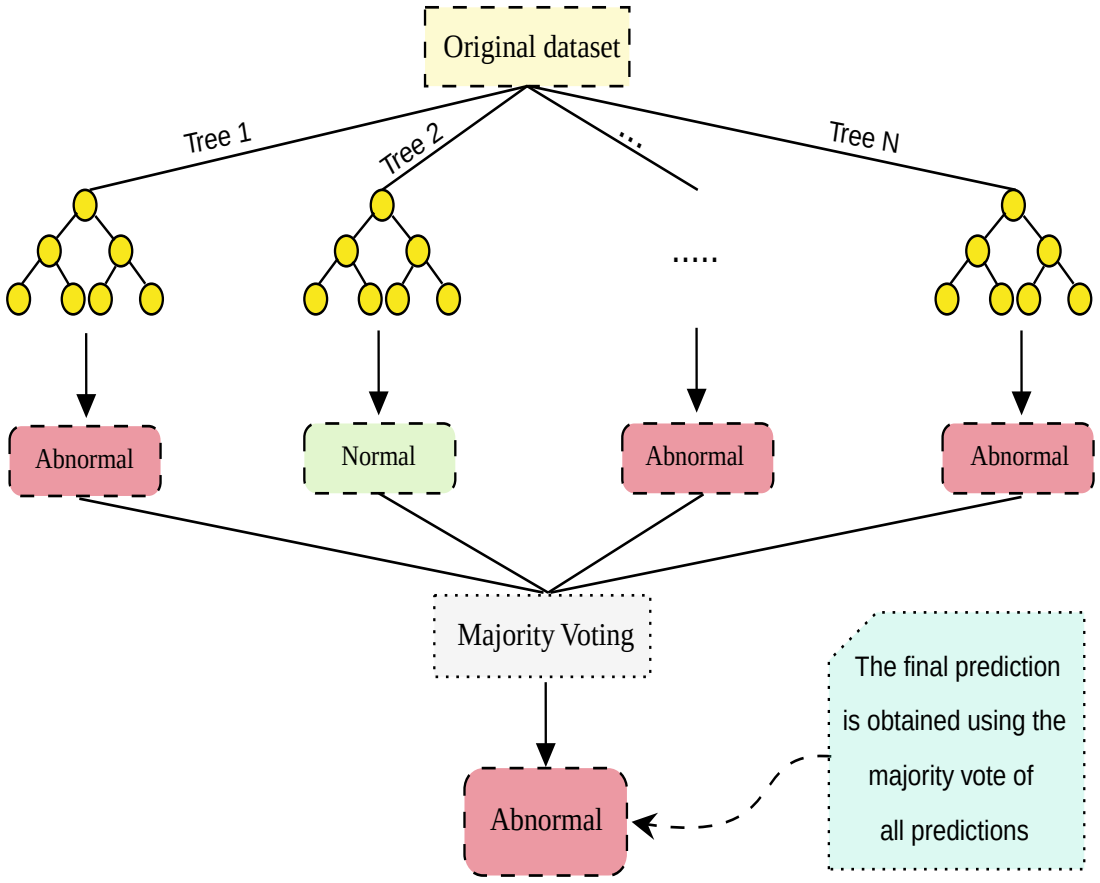


Figure 3.3 – Random Forest classifier (RF)

3.2.2.3 Support Vector Machine

SVM is a supervised machine learning algorithm which can be used for both classification and regression problems. However, it is widely used as it produces excellent performance. SVM tries to find a hyper-plan H that perfectly discriminates between data points of different classes. Assuming that training data are represented as follows: (x_1, y_1), (x_2, y_2), ..., (x_n, y_n) where x_i denotes the input vector and y_i is either 1 or -1 denoting the class label for binary classification. Separating the hyper-plans is of the following form:

$$W.X - b = 0 \tag{3.4}$$

W is a vector that points perpendicularly to the separating hyper-plane and b parameter is used to adjust the margin. The following figure 3.4 illustrates the two class classification problem and its corresponding decision hyper-plans:

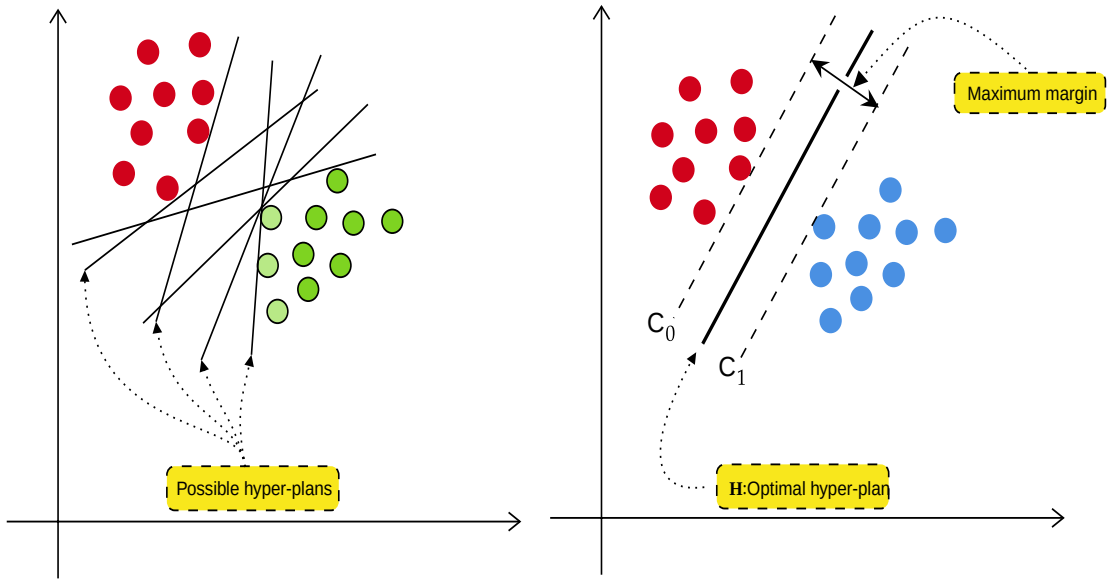


Figure 3.4 – Support Vector Machine (SVM)

Even if, C_0, C_1 and H can separate two classes perfectly, H is the optimal one because it maximizes the distance between H_0 and H_1.

depending on whether the data is linearly separable or not, the SVM can make use of kernels. If data is linear, the construction of hyper-plane is always possible. Otherwise, the usage of kernels in order to map into a higher dimensional feature space is a fundamental element for finding separating hyper-plane.

3.2.2.4 K-Nearest-Neighbor

K-NN is an instance-based non-parametric classifier method. It has been developed first by Evelyn Fix and Joseph Hodges in 1951. Although it is simple,K-NN is powerful and effective in many tasks Huggins and Rudin (2014); Burges (1998). KNN stores all available examples and classifies new examples based on a similarity measure (e.g distance function). Moreover, it does not have a specialized training phase and uses all the data for training during classification (Lazy learning algorithm). A new data-point is classified by a majority votes for its neighbor classes. In other word, each new object is assigned to the most common class amongst its K nearest neighbors. The following steps can provide a further explanation.

1. Step 1: During the first step of KNN, we must load the training as well as test data.
2. Step 2: Choose the value K of the nearest neighbors.
3. Step 3: For each point in the test data do the following:
   (a) Calculate the distance between test data and each row of training data relying on some distance functions such as: Euclidean, Manhattan or

Hamming distance. The most commonly used method to calculate Euclidean distance is between two examples X and Y.

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{3.5}$$

(b) The test data will be assigned a class which is the most frequent among the k training samples nearest to that point.

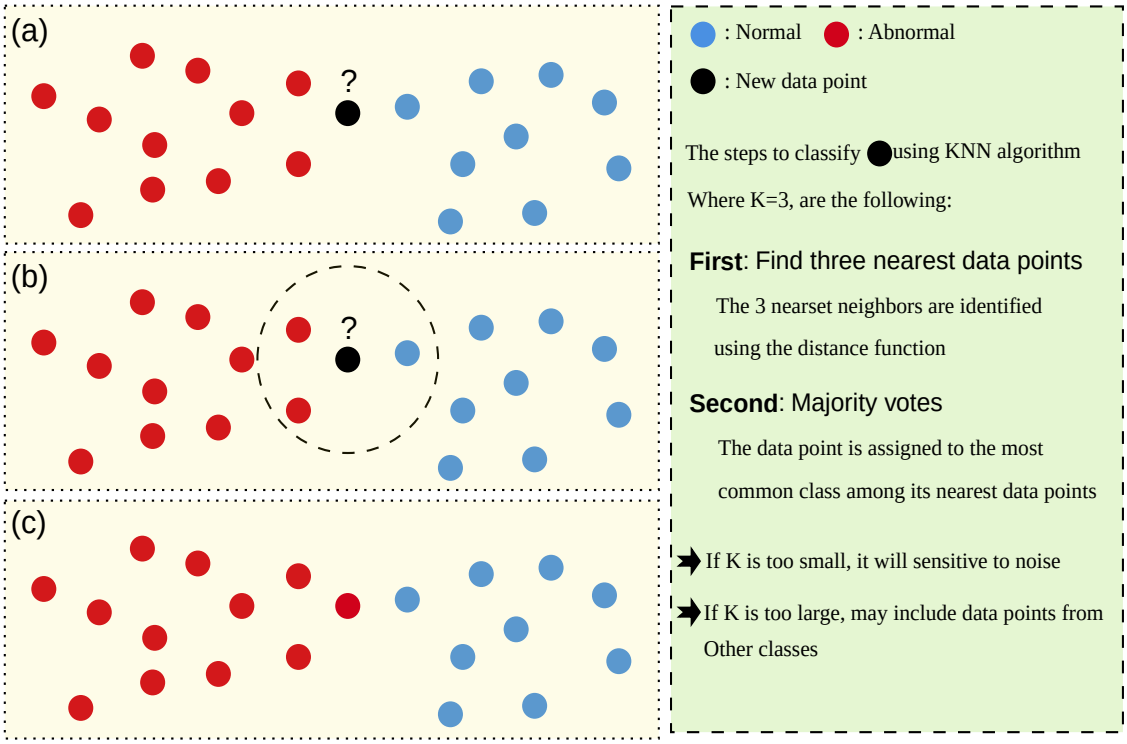


Figure 3.5 – K-Nearest-Neighbor (KNN)

3.2.3 Validation techniques

Evaluating the performance of a machine learning model is an integral component of any data science project. Model evaluation aims to estimate the generalization performance (Accuracy, AUC, Error estimate, etc) of a model on unseen data. Different techniques are introduced in the literature.

3.2.3.1 The train/test/validation split

After cleaning and normalizing all datasets, the most important step before feeding machine learning classifiers is to split the datasets at hand into Train/Test and validation set. This splitting pre-processing is a crucial stage in machine learning because training the model on the entire dataset leads to the likelihood of over-fitting to the training set. This process results in less generalization ability and un-reliable biased results. In order to construct a machine learning model, we generally split the data into three subsets: train, validation (development set),

and test subsets. The training set is employed to train the model; the validation set is used for model selection while the test set is holdout for the final model evaluation. The familiar way to split the data is the following: 60% of the data for training, 20% for validation and 20% for test set. Setting up the training, development (dev) and test sets has a huge impact on productivity. It is important to choose the dev and test sets from the same distribution and it must be taken randomly from all the data. In the modern era of machine learning and deep learning when we are working with much larger datasets, the know-how old guidelines of data splitting have changed. For bigger datasets ($\ggg$1M examples), the dev and test set can have around 10,000 examples each for instance (only 1% of the total data).

3.2.3.2 Holdout technique

The purpose of holdout evaluation is to test a model on different data than it was trained on, which may provide an unbiased estimate of a classifier performance. In this technique, the data is randomly divided into:

- Training set is a subset of the dataset meant to train the model.
- Validation set is used to check the performance of the trained model, parameter-tuning and for model selection.
- Test set is holdout for final evaluation.

The holdout technique is useful because of its speed, simplicity and flexibility. However, this technique is often associated with high variability since differences in the training and test dataset can result in huge differences in the estimate of performance especially when the distribution of data on training is totally different from the distribution of validation/test sets.

3.2.3.3 Cross validation (CV)

Cross Validation is the well-known validation technique; it divides the whole dataset into training and testing set where the test set (unseen by the classifier) is used to check the performance of the model. Several types of cross-validation are proposed in the literature. The most common type is k-fold cross-validation. It divides the whole dataset into k subsets approximately of the same size. k-1 subsets are merged into a training subset and the remaining subset is left out for final evaluation of the model. The evaluation procedure is repeated for different subsets k times. This results in k trained models. The cross validation performance is eventually computed as the mean average of all estimated performance of each model. The following figure 3.6 illustrates the cross-validation procedure with k= 5:

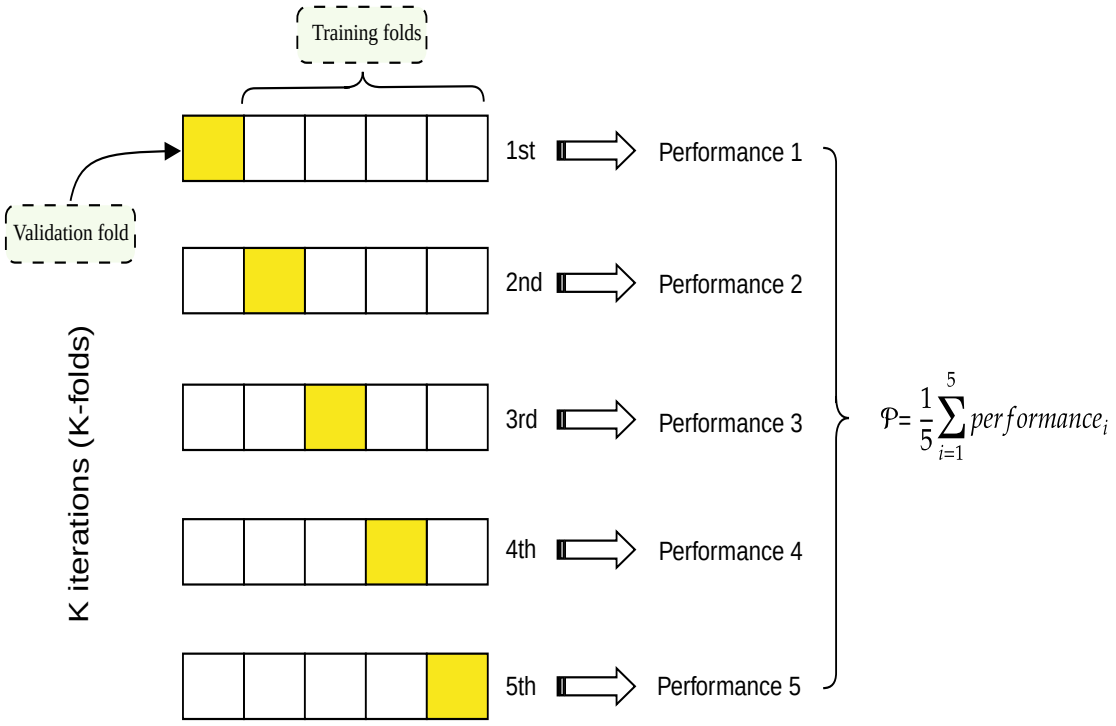


Figure 3.6 – K-folds cross-validation evaluation method with k=5 folds.

For unbalanced datasets where the majority of instances sometimes have the same class, using k-fold cross-validation is not a good choice since it is probably that some generated subset will have the same class for all examples. As a result, the predictor will results in biased results. Another version of k-cross-validation is leave one out cross-validation (LOOCV) where k equal to the number of instances in the dataset. In fact, each sample is used to test the performance of the model and the rest of samples are meant to train the model. For large datasets (large M), LOOCV is impractical since the evaluation process will repeat M times. In addition, it is very sensitive to outliers in data since the model is tested against just one data example.

StratifiedKFold is another variation of cross-validation technique. It is an excellent alternative when dealing with unbalanced datasets since it splits the entire dataset into subset of approximately equal class distribution.

3.2.3.4 Statistical tests

When performing several executions of a given feature selection method, different results are generated (e.g. after applying a k-fold cross validation). In this situation, statistical tests may be performed to check if there are significant differences among the medians for each method. In this thesis, the most used statistical methods is standard deviation since it describes how spread out the values are. A low standard deviation means that most of the generated results are close to the mean (average). A high standard deviation means that the values are spread out over a wider range. It could be helpful to indicate how much the obtained results are across all iterations.

Table 3.2
Confusion matrix representation

Truth	Prediction	
	Positive	**Negative**
Positive	True positive (**TP**)	False Positive (**FP**)
Negative	False Negative (**FN**)	True Negative (**TN**)

3.2.4 Evaluation metrics

3.2.4.1 Confusion Matrix

Model evaluation metrics are indispensable to assess model performance. The choice of evaluation metrics depends on the machine learning task at hand (such as classification, regression, clustering, community detection etc). Supervised learning tasks such as classification and regression constitutes a majority of machine learning applications. In this Thesis work, we focus on metrics for binary classification tasks.

The performance of the classifier can also be evaluated using the so-called Confusion Matrix which is a table of two possible outcomes as is illustrated in the table 3.2.

Let's define the basic concepts of confusion matrix:

- **True Positive (TP)**: for correctly predicted event values.
- **True Negative (TN)**: for correctly predicted no-event values.
- **False Positive (FP)**: for incorrectly predicted event values.
- **False Negative (FN)**: for incorrectly predicted no-event values.

Using the confusion matrix, several standard evaluation metrics can be defined and computed for binary classification.

3.2.4.2 Basic evaluation metrics:

Accuracy

For what fraction of all instances is the classifier's prediction correct (for either positive or negative class)? More formally, using the confusion matrix, the accuracy metric can be defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.6}$$

For unbalanced datasets, accuracy metric is not recommended as the majority of instances have the same class.

Classification error (1-Accuracy)

For what fraction of all instances is the classifier's prediction incorrect? The Classification error can be defined as the following:

$$Classification_{error} = \frac{FP + FN}{TP + TN + FP + FN} \tag{3.7}$$

Precision

What fraction of positive predictions are correct? Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.

$$Precision = \frac{TP}{TP + FP} \tag{3.8}$$

Recall

Also known as True positive rate (TPR), sensitivity or probability of detection. It is the ratio of correct positive predictions to the total positives instances. In other words, what fraction of all positive instances does the classifier correctly identify as positive? Formally, Recall is defined as:

$$Recall/TPR/Sensitivity = \frac{TP}{TP + FN} \tag{3.9}$$

Specificity

It is defined as the following:

$$Specificity = \frac{TN}{TN + FP} \tag{3.10}$$

Area Under the ROC Curve (AUC-ROC)

The area under the ROC curve measures the performance of the classifier at various thresholds settings. As illustrated in figure 3.7, the ROC curve (Receiver Operating Characteristic) plots the **true positive rate (TPR)** on the y-axis versus the **false positive rate (FPR)** on the x-axis.

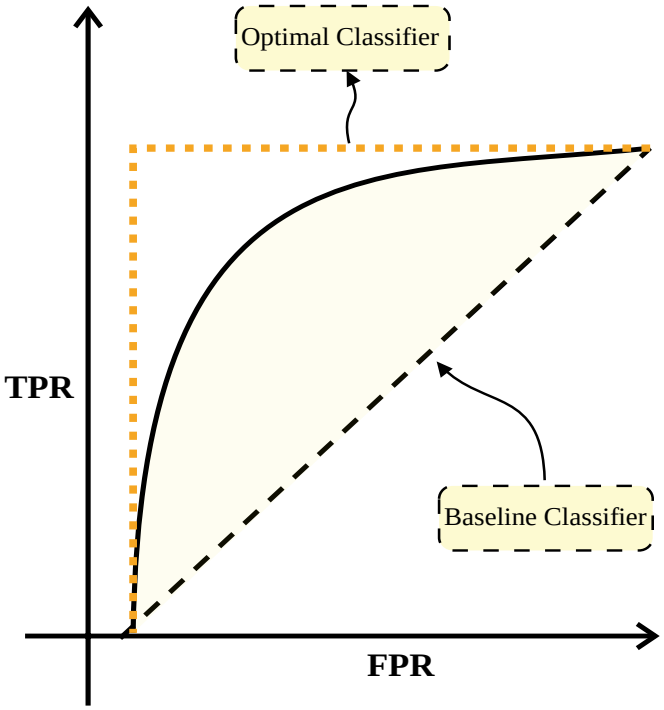


Figure 3.7 – ROC-AUC curve. The AUC represents the degree of measure of separability. The higher the AUC, the good classifier is at classifying data examples.

Root Mean Square Error (RMSE)

RMSE is frequently used metric. It measures the difference between the predicted value by the classifier and the true value. **RMSE** is defined as follows:

$$RMSE = \sqrt{\frac{1}{m}\sum_{1}^{m}(Predict(i) - True(i))^2} \tag{3.11}$$

Where $Predict(i)$ denotes the prediction probability of instance i and $True(i)$ denotes the actual probability.

3.3 Software tools

The experiments performed in this thesis were executed using the software tools Paython, Weka and R, which are described in the following paragraphs:

1. **Matlab** (Matrix Laboratory)Matlab (2013) is a numerical computing environment, well known and widely used by scientific researchers. It was developed by MathWorks in 1984. Matlab allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

2. **Weka** (Waikato Environment for Knowledge Analysis) Hall et al. (2009) is a set of machine-learning algorithms for data mining problems. The algorithms can be either applied directly to a dataset or called from your own Java code. Weka contains tools for data preprocessing, classification, regression, and visualization. It is also well suited for developing new machine learning algorithms.
3. **Python** Van Rossum and Drake Jr (1995) is powerful programming language, easy to learn. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. The Python interpreter and the extensive standard library are freely and publicly available in source or binary form for all major platforms from the Python Web site:`https://www.python.org/`.

3.4 Summary of the chapter

In this chapter, the quintessential focus of this pinpointed discussion has been emphatically placed on the properties of the evaluation methodology commonly applied in feature selection. First, we have presented the evaluation methodology used in this thesis. Second, we have described the employed datasets to check the validity of the proposed FS. In addition, we have discussed the well-known evaluation and validation techniques. As to the end of the chapter, the software tools used to implement the previously discussed FS methods have been presented.

The following chapters (4, 5, 6 and 7) will be devoted to the proposed feature selection methods developped to efficiently identify and select the optimal features. The next chapter will discuss two proposed FS methods. The first one is an hybrid filter-wrapper method and the other one relies on graph representation to quantify the reliability of selected features.

4

A Graph-Based Approach for Feature Selection

“

“The key to artificial intelligence has always been the representation”.

Jeff Hawkins.

Chapter 4

A Graph-Based Approach for Feature Selection

Contents

In this chapter, two feature selection methods are proposed. The first one is a hybrid filter-wrapper method based on pairwise feature evaluation. It combines the speed up of filters and the high performance of wrapper. However , it fails at capturing and identifying high-order interactions especially when dealing with complex real-world tasks. For this reason, a graph feature selection method is proposed where feature space is modelized using a graph representation. Each feature is represented by a node. A modularity function is applied to split the whole graph into communities and the best subset of features is selected among the identified ones. This proposal shows its effectiveness in selecting the proper features as it is evaluated on eight benchmarking datasets which are previously discussed in **chapter 3**.

4.1 Introduction

Feature selection is a very important pre-processing technique in machine learning and data mining. It aims to select a small subset of relevant and informative features from the original feature space which may contain many irrelevant, redundant and noisy features. Feature selection usually leads to better performance,

interpretability, and lower computational cost. In the literature, FS methods are categorized into three main approaches: Filters, Wrappers, and Embedded. Given the choice between wrapper and filter approaches, wrappers are often preferred when a learning algorithm has already been selected, because wrappers rely on learning algorithm to select feature subsets that perform well with this particular algorithm. A serious drawback of wrappers is their high computational requirement and their possibility to over-fit. Filter methods, by evaluating features one by one, do not share this disadvantage, and are thus generally much faster than wrappers. There is, however, a drawback to considering every feature by itself: By not pair-wisely evaluating features, filters fails dealing with correlations and redundancies.

In the first contribution of this chapter, we propose a hybrid filter-wrapper approach to feature selection. We use a learning algorithm to evaluate the performance of pairs of features. This constitutes the wrapper part of our proposal. We then rank the features based on the matrix of pairwise performance which constitutes the filter part of our approach.

Based on the limitations of the first proposed method, we have introduced a new feature selection method called graph feature selection (**GFS**). The main steps of **GFS** are the following: first, we create a weighted graph where each node corresponds to each feature and the weight between two nodes is computed using a matrix of individual and pairwise score of a Decision tree classifier. Second, at each iteration, we split the graph into two random partitions having the same number of nodes, then we keep moving the worst node from one partition to another until the global modularity is converged. Third, from the final best partition, we select the best ranked features according to a new proposed variable importance criterion. The results of **GFS** are compared to three well-known feature selection algorithms using nine benchmarking datasets. The proposed method shows its ability and effectiveness at identifying the most informative feature subset.

4.2 Pairwise feature evaluation

In the first proposed algorithm of this chapter, we introduce a new feature selection algorithm based on evaluating features on pairs. Some algorithms of feature selection based on pairs are already proposed in the literature. Trand hellem and al Bø and Jonassen (2002) present two alternative methods for selecting a subset of features based on evaluating pair of features. One exhaustive called " All pairs ", in which the authors consider all pairs of features. After calculating pair t-scores (conditional t-score) for all possible pairs, they select the top-ranked pairs; then all pairs containing any of these two features are removed from the remaining list of pairs. The highest scoring pair from the remaining list is chosen, and so on. This method is computationally expensive when the number of features is important. The second method called (greedy pairs) is faster; it ranks features individually using t-score, subsequently, this procedure firstly selects the best features ranked by t-score, then it seeks to find the feature that together with an already selected feature maximizes the pair t-score, these two features are removed from

the list of the remaining features, and so on. The main drawback with the t-score approach is that the data are supposed to be normally distributed. In Dreiseitl and Osl (2009), Stephan Dreiseitl1 and al. propose a hybrid filter-wrapper algorithm. They start by choosing the feature with the highest individual discriminatory power using Logistic Regression and AUC metric, then they incrementally rank features by choosing as next feature the one that achieves the highest AUC in combination with an already chosen feature. We found that this method has three drawbacks:

- First, using only the top ranked feature as a starting point can lead to a sub-optimal subset since it does not guarantee that the first obtained pair of features is the best pair among all the available ones.
- Second, the process of feature selection does not take into account all the previously selected features but only one of them, hence, we can maximize the score of the obtained pair but decrease the overall score of the whole set of chosen features. Moreover, this process allows the selection of redundant features or highly correlated features with the already chosen features. Let A be the chosen set of features and B, C, D the remaining set of features. Suppose that B and C are redundant and that the pair (A, B) have the maximum score. After B have been chosen, C will be chosen at the next iteration as it will also maximize the score (A, C).
- Third, the use of the Logistic Regression (with a 10-folds cross validation) shows to be extremely slow in practice using datasets of moderate size.

4.2.1 Proposed method

In this section, we propose a new approach which avoids the drawbacks of the previous works and is much faster to run. We, first, introduce our features selection algorithm and then we will motivate each step of it. Suggested pairwise feature selection method is processed using two folds cross validation. The score is determined as the average AUC of the decision tree. At each phase, the selected feature should satisfy two criteria. The first one is about avoiding the selection of redundant features and the second one is about selecting the best features among those satisfying the first criterion. The following steps give a more precise description of the algorithm:

1. **Step 1: Rank all features individually.**
   The process of ranking of each feature is done by using the AUC score of Decision Tree model trained only on this feature. In this procedure, features are ranked from the most to the least relevant according to the values of AUC score.
2. **Step 2: Compute pairwise AUC score.**
   Calculate the AUC score of all pairs of k best features with all the features and store them in an $N \times N$ symmetric matrix of pairwise scores where the diagonal is initialized by the individual AUC score of all features.
3. **Step 3: From the best pair of features, select the feature F_s with the highest individual score as the starting feature.**

- S=F_s initialize S by the best first feature.
- R=remaining features

4. **Step 4: Iterate the following until we reach the desired number of features.**
   The next selected feature f_i should obey the two following criteria successively:

   (a) The pair AUC score of the current and already selected features should be greater than the individual AUC score of already selected features. Formally we can write :

$$\begin{cases} C = \{f_i\} & if AUC(F_i, F_s) > AUC(F_s) \forall\ F_s \in S\ and \forall\ f_i \in R \\ C = R & otherwise \end{cases} \tag{4.1}$$

   (b) The best feature should also maximize the following score:

$$argmax_{i \in C} \sum_{s \in S} (AUC(f_i, f_s) - AUC(f_s)) \times AUC(f_s) \tag{4.2}$$

   (c) Compute all pairs of the selected feature.

After computing the scores of individual features, we compute the pair scores of K best features in step 2. Instead of computing all the pairs, which is time consuming. We choose K to be the square root of the number of features. The reason behind is that the best features will highly have the best pair scores among all existing pairs. K can also be set manually by the user to override the default choice. After choosing the starting feature, we iterate over all the features to select the best next one. In Bø and Jonassen (2002), the process of selection is only based on score maximization of the current feature with one of the already selected features, but this can lead to the selection of redundant features. To avoid this problem, at each iteration we create a set C of candidate features 4.1, this way we achieve two objectives: the candidate feature is not redundant and it interacts well with all selected features not with just one of them. If no candidates are found, we allow C to be the remaining set R.

Next step, for each candidate we compute a score using the equation 4.2, the selected feature is the candidate maximizing that score among all the candidates. The equation 4.2 has two parts: $AUC(F_i, F_s)$ is the increase in the AUC score when f_i interacts with f_s. But since an increase of an AUC score from 0.8 to 0.81 is more significant than an increase from 0.6 to 0.61, a weighting term should be introduced to take the importance of the already selected features into account.

4.2.1.1 Evaluation

The proposed method has been evaluated using just two folds cross-validation instead of ten folds in Dreiseitl and Osl (2009), the reason for that choice will be explained later. At each iteration, the quality of features ranking is determined using AUC score of the Decision tree classifier with depth equal to three. The

depth parameter is chosen to be small as we do not intend to build the best model for each pair, which needs an exhaustive search among all the hyper-parameters of the decision tree but rather evaluate the relative performance of each pair of features. We choose to use the AUC metric because it is more suitable for evaluating the performance of classifiers on unbalanced data sets. The final subset of selected features is evaluated using a Random Forest with a grid search strategy for the hyper-parameters. The AUC score is computed using the out of bag (OOB) score of the random forest model.

4.2.1.2 Experiments

This section illustrates the evaluation of our method in terms of AUC score in order to detect how well our method is.

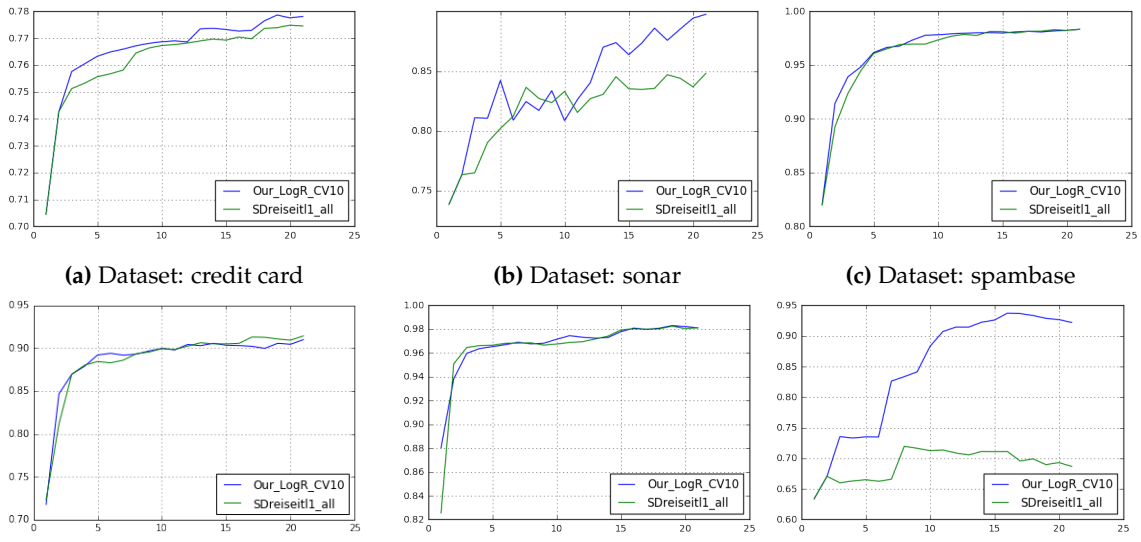
Our method is evaluated in two parts:

i. Evaluation of the selection process itself without the involvement of any parameters as the first benchmarking.

ii. Evaluation of the choice of using some parameters (Classifier, number of folds cross validation) as the second benchmarking

First experiment

The first benchmarking is about comparing our approach with SDreiseitl1 et al Dreiseitl and Osl (2009) to figure out the strength of our feature selection method. We set the parameters as described in Dreiseitl and Osl (2009): the classifier used to evaluate pair of features is the Logistic regression and the cross validation is done with 10 folds.



(a) Dataset: credit card **(b)** Dataset: sonar **(c)** Dataset: spambase

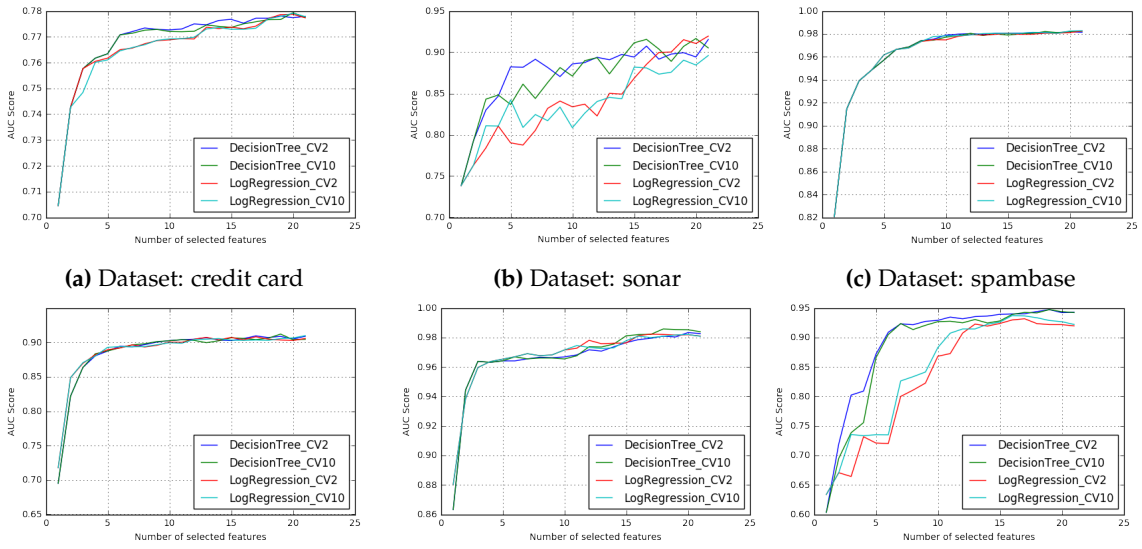
(d) Dataset: ds1.100 **(e)** Dataset: musk **(f)** Dataset: madelon

Figure 4.1 – Comparison of our feature selection approach (blue curve) and $SDreiseitl1_{al}$ approach (green curve). The x-axis is the number of selected features. The y-axis is the performance in terms of AUC score using LR and 10 folds cross-validation.

The illustrated results in Figure 4.1 show that our method outperforms SDreiseitl1 et al pairwise method in most datasets, especially Credit card client, Sonar and madelon datasets. Maximizing only the score of one of the selected features Dreiseitl and Osl (2009) may work well on some datasets but clearly shows its limits on other datasets especially on those with very few instances (208 instances in sonar) or a high number of features (500 features in madelon). Our selection approach based on optimizing all the interactions with previously selected features shows to be more consistent across all the datasets even those with high number of features or low number of instances.

Second experiment

In the second benchmarking, we will study the impact of the parameters (classifier, number of folds) on the quality of the selected features. We will use Decision Tree with 2 folds cross validation ($DecisionTree_{CV2}$), Decision Tree with 10 folds cross validation ($DecisionTree_{CV10}$), Logistic regression with 10 folds cross-validation ($LogRegression_{CV10}$) and Logistic regression with 2 cross-validation ($LogRegression_{CV2}$).



(a) Dataset: credit card **(b)** Dataset: sonar **(c)** Dataset: spambase

(d) Dataset: ds1.100 **(e)** Dataset: musk **(f)** Dataset: madelon

Figure 4.3 – Impact of the parameters on our feature selection algorithm. The x-axis is the number of selected features. The y-axis is the performance in terms of AUC score using LR and DT with 2-folds and 10-folds cross-validation. The performance of our FS method ($DecisionTree_{CV2}$) using just 2-folds cross validation outperforms other methods.

As we can see in the Figure 4.3, the classifier $Decisiontree_{CV2}$, outperforms other classifiers in terms of AUC score in three datasets: Credit Card client, Sonar and madelon. Using the two datasets spambase and ds1.100, $Decisiontree_{CV2}$ has similar score compared to the other classifiers. For dataset Musk, $Decisiontree_{CV2}$ performs well at the start, then AUC score decreases a little after the sixth selected feature. Therefore, it is clear that the use of the classifier $Decisiontree_{CV2}$ in our pairwise method is the best choice.

Regardless of the number of folds used in the experiments, we can see that Decision tree outperforms Logistic Regression. This is due to the fact that Logistic Regression is a linear model that can not handle nonlinear interactions that may exist between features. Therefore, we recommend the use of Decision trees with depth equal to 3 over Logistic Regression as it gives better performance and executes much faster. We, also, recommend the use of a 2 folds cross validation strategy which is very advantageous in terms of execution time without loss of performance.

4.2.1.3 Limitations

In this first method, we have proposed a hybrid filter-wrapper feature selection method that combines advantages of both wrapper as well as filter approaches, by using Decision tree and the area under the ROC curve (AUC) to evaluate pairs of features. The experiments were designed to evaluate the performance of the proposed approach with six benchmark datasets. The results indicate that the selected subset of features by the proposed approach has a good classification performance especially for datasets of moderate size. For madelon dataset which contains 500 features, the proposed pairwise method P-FS quickly achieves its highest AUC score with only 16 features then, P-FS performance starts to downgrade. This is not surprising since relying just on pairwise interactions leads to ignore third or higher-order feature interactions that cannot be found looking only at pairs of features. To capture beneficial feature interactions of high-order, we suggest to use graph representation where all possible interactions are taken into account.

4.3 Graph feature selection

The second method of this chapter is the proposal of a new feature selection approach based on graph data representation counting on community detection in order to reduce the search space and select the best performing feature subset. The choice of the graph representation is made relying on the fact that graphs are powerful and many-sided data structure which allow us to easily represents real world datasets and the interactions between different type of features.

4.3.1 Graph representation

A graph, or a network, G is simply a couple of collections (N; E), where N is a set of nodes and E is a set of edges, so that each edge being a pair of nodes in N. Nodes are also referred to as vertices. Edges are also referred to as links or connections. If each edge is an unordered couple of vertices, the edge is undirected. Thus, the graph is an undirected graph. Otherwise, if each edge is an ordered couple of vertices, the edge is directed from one vertex to the other. So, the graph is a directed graph. In this case, an ordered couple of vertices $(v_1; v_2)$ is an edge directed from vertex v_1 to vertex v_2. If each edge has an associated numeric value called weight, the edge is weighted, and the graph is a weighted graph. Figure 4.5 shows four models of graphs, including an undirected graph, a directed graph, a weighted undirected graph and a weighted directed graph. In this contribution,

we have used the weighted undirected graph to represent the underlying interations between features relying on the fact that the pairwise score between two features $(F_1; F_2)$ is the same as the score of $(F_2; F_1)$.

Modeling feature selection problem by graphs or networks offers an ideal tool to visualize how the features and variables interact to properly classify a given dataset. In our proposal graph theoretical based approach for feature selection, a feature selection problem is conceptualized as a graph G(N; E), where N is the set of nodes denoting features/variables and E is the set of edges denoting interactions or the pairwise AUC score in our proposed Graph feature selection (GFS). The interaction between two nodes u and v is represented by the edge $e_{u;v}$ of the graph.

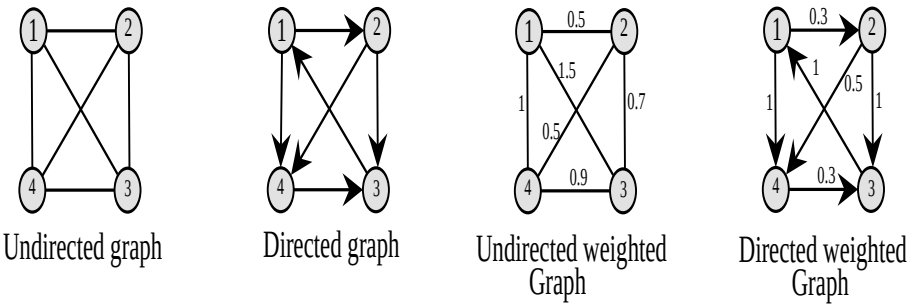


Figure 4.5 – Graphs' examples

4.3.1.1 Community detection

Undoubtedly, the structure of interactions among features/variables in real-world datasets is complicated Girvan and Newman (2002); Newman (2003). Therefore, modeling and analyzing them using graph structure is a crucial key in discovering and selecting the informative features among thousand of irrelevant/redundant ones. The graph representation habitually displays community structure (sometimes referred to as clustering). A network is said to display such structure if its nodes can be divided into either overlapping or disjoint groups such that the number of internal edges exceeds the number of external edges by some reasonable amount. An internal edge is a link connecting two nodes belonging to the same groups, whereas an external edge is a link connecting two nodes of different groups. Networks exhibiting a community structure may frequently display a hierarchical community structure as well. This signifies that the network may be partitioned into a small number of large communities, and each of these may be partitioned into many smaller communities. Various algorithms that reveal the hierarchical community structure of graphs are introduces in the community detection literature Akachar et al. (2021).

4.3.1.2 Modularity

The Modularity algorithm which has been introduced by Newman and Girvan Newman and Girvan (2004), is a measure that evaluates the goodness of a network's division into communities. The goal of the modularity function is that a random graph is not supposed to have a community structure, so the probable

presence of communities is exposed by the comparison between the real density of edges in a subgraph (a community candidate) and the density one would expect to have in a randomized version of the same sub-graph. The randomized version of the subgraph is a portion of a randomized version of the complete input network referred to as a null model, i.e., a representation of the original graph having some of its fundamental characteristics yet without community structure. Before explaining the main idea of this function, we need to define some concepts related to it.

Definition (Adjacency matrix): The adjacency matrix A_G for an undirected and weighted graph G of N nodes is a square matrix of ones and zeros that indicate whether pairs of nodes are adjacent or not in the graph. The adjacency matrix elements $a_{i;j}$ are one when there is an edge between two nodes i and j, and zero when there is no edge between them. More formally, for an undirected and weighted graph $G = (N, E)$ where $N = \{1, 2, ..., n\}$ and $E = \{e_1, e_2, ..., e_m\}$, the adjacency matrix A_G of G is an $n \times n$ matrix: $AG = [a_{i,j}]_{1 \leq i;j \leq n}$ where:

$$\{1 \quad if \;\; w_{i,j} > 00 \;\; otherwise \tag{4.3}$$

where $w_{i,j}$ is the pairwise score between two features F_i and F_j

Definition(Node degree): In undirected and weighted graph $G = (N, E)$, the degree of a node is the number of edges connected to the node. In terms of the adjacency matrix AG, the degree for a node indexed by i is: $d(i) = \sum_{j \in N} a_{i,j}$

Consider a graph $G = (N; E)$ with $|N| = n$ is the number of nodes and $|E| = m$ is the number of edges. Assume we are given a community structure P for the graph where $\mathcal{P} = \{C_1; C_2; ...; C_{nc}\}$ is a partition of N into communities. As mentioned above, from the view of modularity, the quality of a community division of a network can be asserted by the difference between the actual fraction of edges within communities and that expected fraction of edges in an equivalent random network. Modularity can then be written as follows:

$$\mathcal{Q}(\mathcal{P}) = \frac{1}{2m} \sum_{ij} (A_{ij} - P_{ij}) \delta(C_i, C_j) \tag{4.4}$$

Where the sum runs over all pairs of nodes, A is the adjacency matrix, m the total number of edges of the graph, and P_{ij} represents the expected number of edges between nodes i and j in the equivalent random graph,P is a partition of the graph into communities, C_i and C_j are the communities of nodes i and j respectively, and $\delta(C_i, C_j) = 1$ if nodes i and j are in the same community ($C_i = C_j$), and 0 otherwise.

4.3.2 Proposed method

The main steps of the proposed algorithm are the following: first, we rank all features individually regarding to the AUC score of decision tree. Second, we compute all pairs of features and store them in an N×N matrix. Third, we construct a weighted graph G where each feature is represented by a node and the edge

between two nodes corresponds to the pairwise score. Fourth, after constructing the graph G, we split the nodes of the whole graph into two random partitions which contain the same equal number of nodes. Fifth, we move the node (feature) of lower fitness from one partition to another and compute the modularity $\mathcal{Q}$. The step 4 and 5 are repeated until convergence ($\mathcal{Q}$ is maximum). The eventual finalization of this splitting and moving procedure, we result in several groups of features. At this stage, we propose a new criterion to select the most informative and relevant features.

For an in-depth analysis of the proposed graph feature selection algorithm, we suggest the following pseudo-code:

1. **Compute individual score.**
   All features are ranked in decreasing order of AUC score of the decision tree.
2. **Compute the pairwise score.**
   We calculate the score of pairs of features and we store them in an N×N matrix where the diagonal is initialized by the individual score.
3. **Create a weighted graph G.**
   We construct a weighted graph G where each node corresponds to each feature, and the pairwise score between two features is used as the weight of the edge between two nodes.
4. **Split the graph G.**
   After the post-graph G construction step, we split the whole nodes of this graph into two random partitions having the same number of nodes.
   - Moving the node (feature) with the lower fitness from the current partition to another and calculate the local modularity q. This step is repeated until K nodes are moved or when the local modularity q cannot be improved any further.
   - The fitness of each node is calculated as the sum total of all the weights of the outgoing edges derived from that node.
   - Each partition can be viewed as sub-graph.
5. **Repeat the splitting process until convergence.**
   We keep splitting the node of each sub-graph until the global modularity of the initial graph G is maximum. We defined the convergence of our algorithm as follows: Our algorithm is converged when the difference between the global modularity of two successive iterations is smaller than a given values C.
6. **Feature selection**
   From the final communities (partitions), we try to select the most informative and useful features meant to build a good model. The final best features should be abided by the following variable importance criterion:

$$VI = \frac{\frac{NDIP}{TD} + IS}{2} \tag{4.5}$$

Where **VI** is variable importances, **NDIP** is The node degree inside the partition, **TD** is the Total degree of the initial graph G and **IS** represents the Individual score.

After ranking features according to the proposed criterion, the top ranked features are selected as the best subset and the remaining features are discarded and removed.

4.3.3 Starting example

To both illustrate and give more information about this proposed procedure, we apply it on a generated dataset of 7 features.

- **Step 1 : Compute individual and pairwise score**
  A decision tree with depth equal to two, we compute all AUC scores and store them in $N \times N$ matrix as follows:

Table 4.1
Individual and pairwise score using decision tree classifier.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7
F_1	0.5	0.7	0.8	0.51	0.3	0.6	0.59
F_2	0.7	0.4	0.82	0.66	0.9	0.4	0.77
F_3	0.8	0.82	0.6	0.35	0.4	0.6	0.92
F_4	0.51	0.66	0.35	0.3	0.83	0.44	0.55
F_5	0.3	0.9	0.4	0.83	0.2	0.51	0.73
F_6	0.6	0.4	0.6	0.44	0.51	0.7	0.61
F_7	0.59	0.77	0.92	0.55	0.73	0.61	0.52

- **Step 2: Construct a weighted graph G**
  Based on the previously created matrix, we construct a weighted graph G in which each node represents its correspondent feature and the weight of each edge matches the pairwise score between two features.

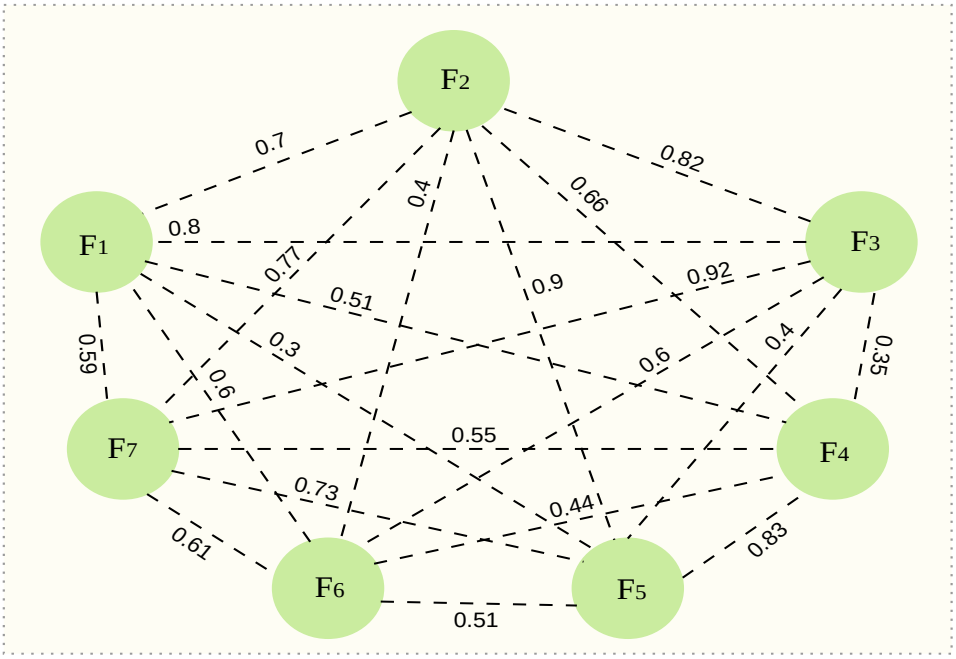


Figure 4.6 – Construct a weighted graph G based on the scores presented in table 4.1.

- **Step 4: Split the graph G**
  We split the graph G randomly into two partitions of the same number of nodes. In our case, the partitions are the following:
  — Partition 1: contains the features F_1, F_2, F_6, F_7
  — Partition 2: contains the features F_3, F_4, F_5

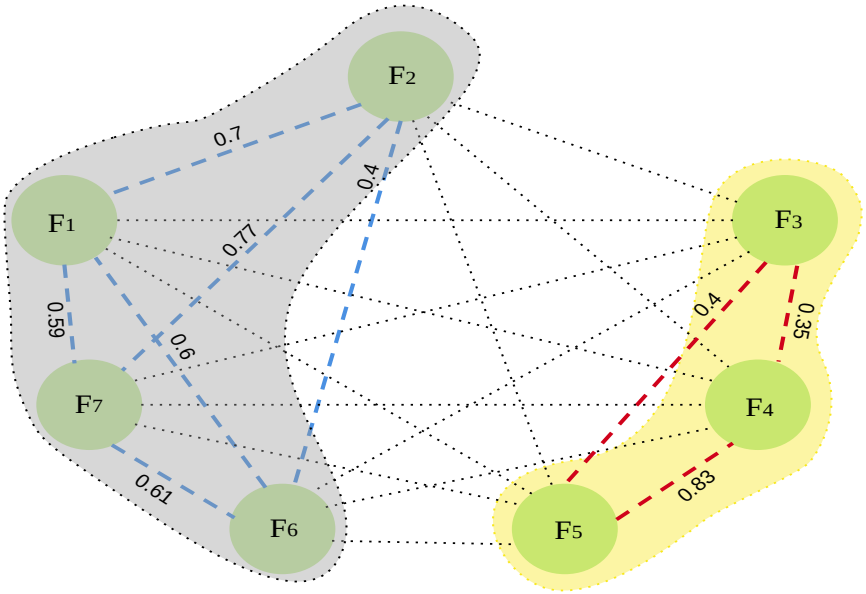


Figure 4.7 – Split the graph G into two random partitions. The first partition includes (F_1, F_2,F_7 and F_6) while the second partition includes (F_3, F_4, $and F_5$).

After splitting the initial graph G, we start moving nodes from one partition to another. The node with the lower fitness should be moved. Let's compute the fitness of each node as discussed previously in the pseudo-code.

Table 4.2
The fitness value of each feature

	F_1	F_2	F_3	F_4	F_5	F_6	F_7
Fitness value	3.5	4.25	3.89	3.34	3.67	3.16	4.17

From the computed finesses, it is obvious that the node F6 must be moved to another partition as it is displayed in the following figure 4.8.

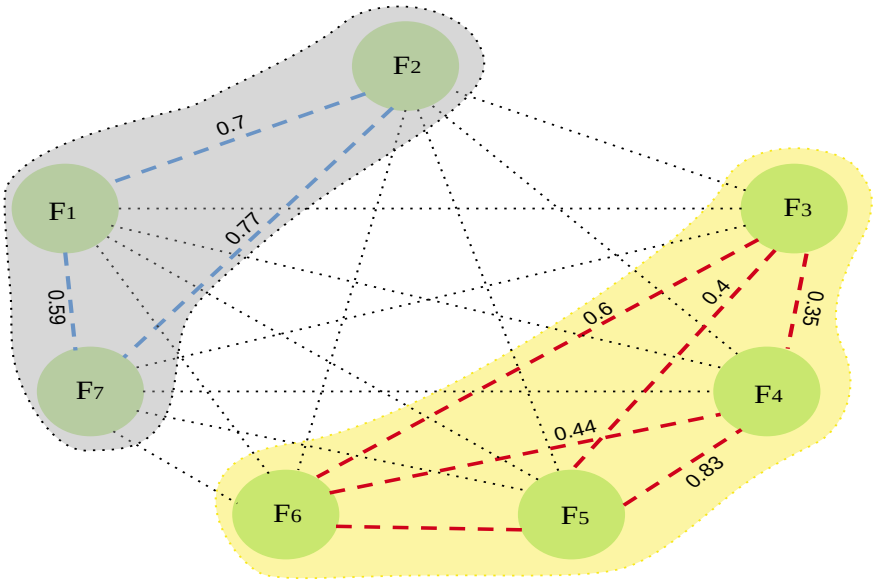


Figure 4.8 – Moving nodes from one partition to another. The node F_6 is moved from the first partition to the second aiming to increase the local modularity q.

As long as the local modularity q does not reach its maximum yet, the moving procedure is repeated until K nodes are moved.

- **Step 5: Repeat the process of splitting and moving until the global modularity $\mathcal{Q}$ is converged.**
  We keep splitting the partitions of the graph randomly until the global modularity is converged which means that the final partitions are the best partitions. In our experiment the algorithm is converged when the global modularity $\mathcal{Q}$ of two successive iterations is smaller than 0.001. The final detected communities or partitions are the following:

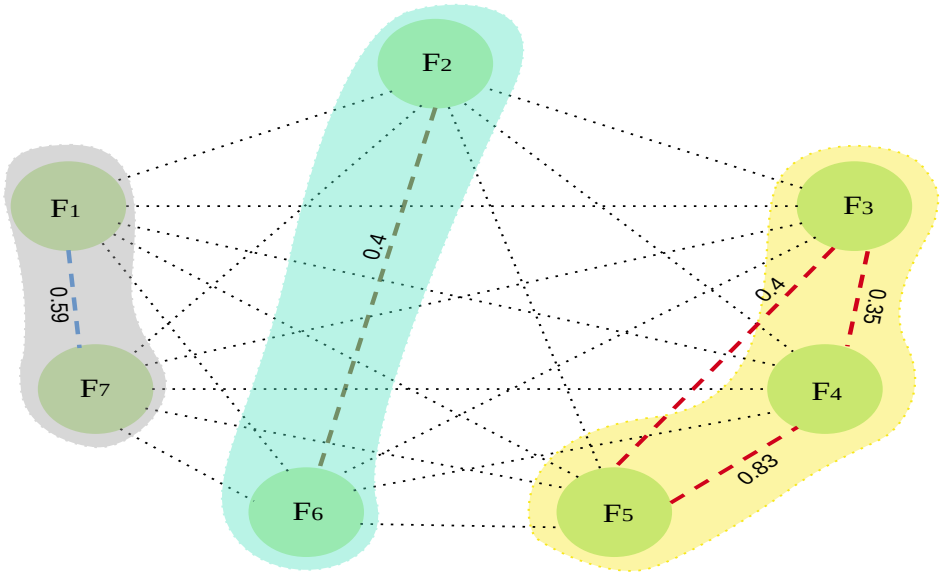


Figure 4.9 – keep splitting the partitions of the graph randomly until the global modularity $\mathcal{Q}$ is converge. At the end of the splitting process, Three partitions are returned.

- **Step 6: Feature selection.** From the final best partitions detected by the proposed algorithm, we will select the best performing subset of features. By applying the proposed criterion of variable importance, here there are the ranking of all features.

  — $VI(F_1) = \frac{\frac{1}{7\times6}+0.5}{2} = 0.262$
  — $VI(F_2) = \frac{\frac{1}{7\times6}+0.4}{2} = 0.224$
  — $VI(F_3) = \frac{\frac{2}{7\times6}+0.6}{2} = 0.324$
  — $VI(F_4) = \frac{\frac{2}{7\times6}+0.3}{2} = 0.174$
  — $VI(F_5) = \frac{\frac{2}{7\times6}+0.2}{2} = 0.124$
  — $VI(F_6) = \frac{\frac{1}{7\times6}+0.7}{2} = 0.362$
  — $VI(F_7) = \frac{\frac{1}{7\times6}+0.52}{2} = 0.272$

  Relaying on the variable importance of all features, the best feature subset selected by the proposed graph feature selection is $[F_6, F_3, F_7, F_1]$ and the remaining features must be discarded.

4.3.4 Evaluation

In this section, an experiment is conducted to evaluate the effectiveness and the efficiency of the proposed graph feature selection algorithm. The performance of our algorithm (**Graph-FS**) is compared to three well-known feature selection algorithms which are: LASSO 2.4.1, RIDGE 2.4.2 and Recursive feature elimination with cross validation (RFECV)Akhtar et al. (2019).

Each dataset is divided into train and test set. The train set is used with two folds cross-validation to compute the weight matrix. The decision tree with depth equal to 2 is trained using just one feature to initialize the diagonal of the matrix (individual score), and we trained the same model using a combination of two features (pairwise score) and store them in the same matrix.

The final feature subset selected by each feature selection algorithm in the comparative study previously conducted (Graph-FS, LASSO, RIDGE and RFECV) is evaluated on the holdout set (test set). For the sake of a fair comparison, the Random forest with a grid search strategy is trained using each generated feature subset, and the AUC score is calculated on the out of bag (OOB) score of the random forest. The choice of AUC score is based on the fact that it is proper performance measurement than accuracy for the classification problem where datasets are unbalanced Hossin and Sulaiman (2015).

4.3.5 Experiments

To test the performance of the graph feature selection at splitting the graph into best partitions, we compute the local modularity q and the global modularity $\mathcal{Q}$ using the graph constructed using the dataset chess which contains 36 vertices. Our algorithm **Graph-FS** detects four partitions after maximizing the global modularity $\mathcal{Q}$ as is shown in the figure 4.10 .After moving each node at each iteration; the local modularity is remarkably increased. When the local modularity cannot be improved any further, the moving procedure should be stopped and then, we try to split the graph G again until no improvement can be achieved in the global modularity.

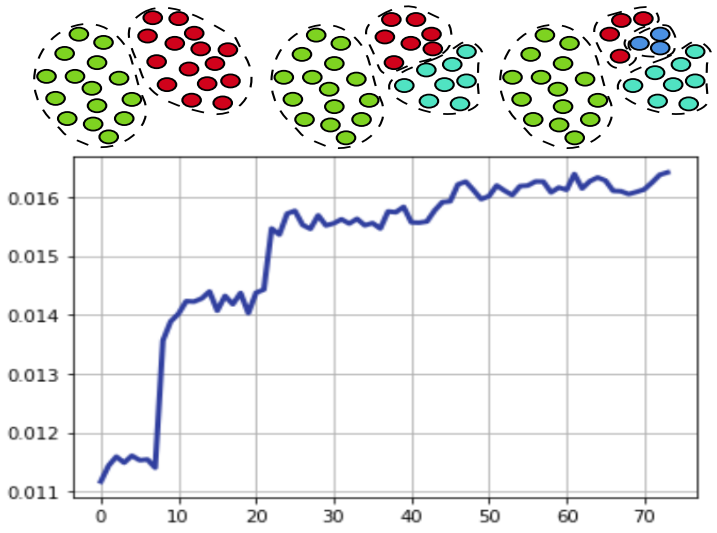
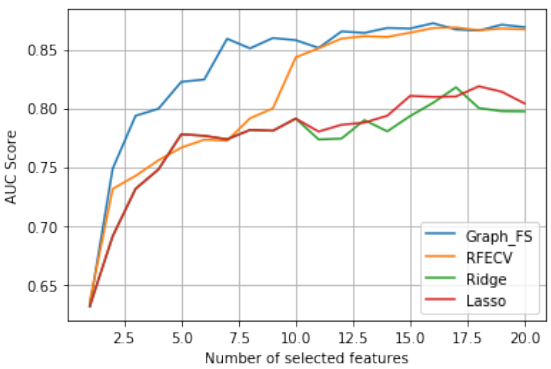
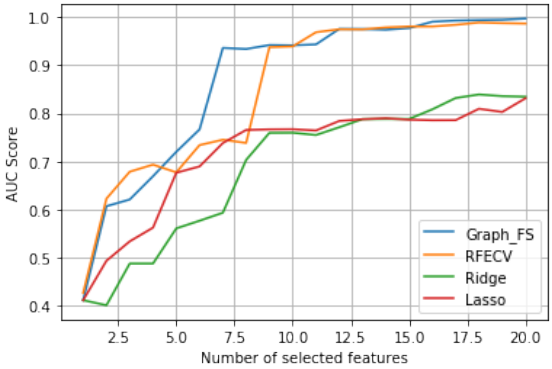


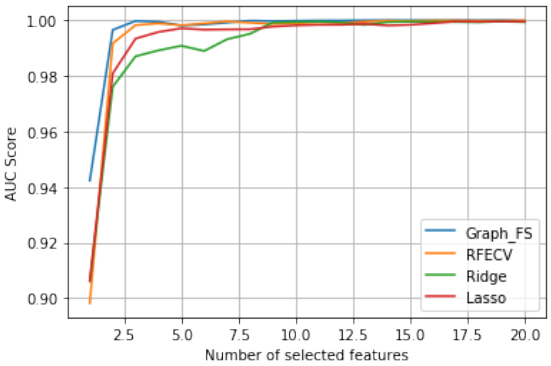
Figure 4.10 – The local q and global modularity $\mathcal{Q}$ on chess dataset. q is used to decide when to stop moving nodes from each partition to another while $\mathcal{Q}$ is meant to decide when to stop the graph splitting process.



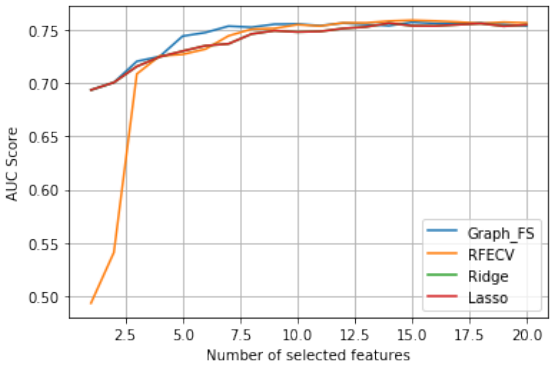
(a) Dataset: ds1.100



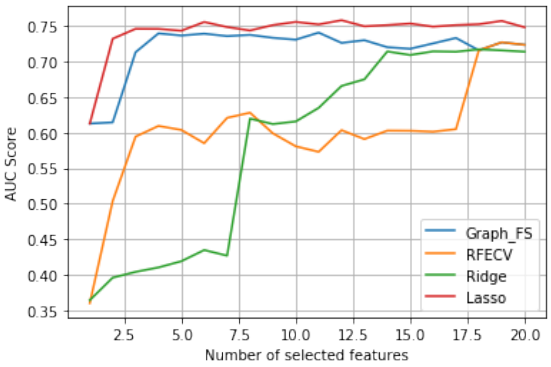
(b) Dataset: chess



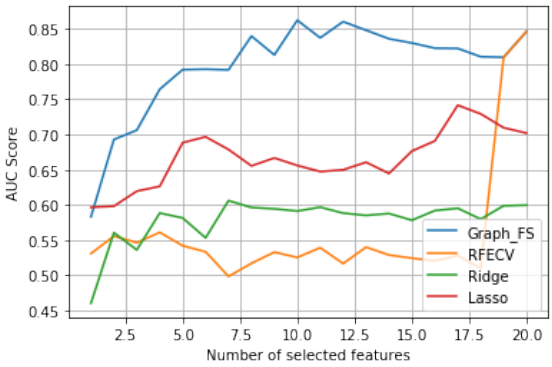
(c) Dataset: Musk



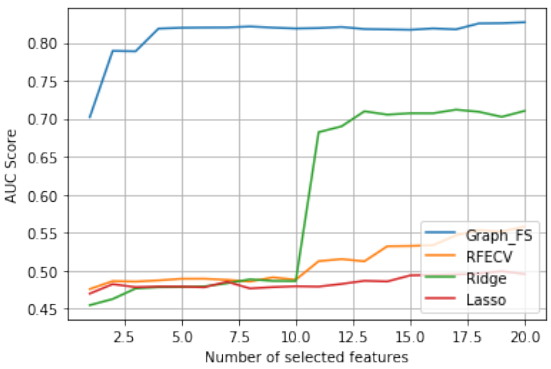
(d) Dataset: credit card



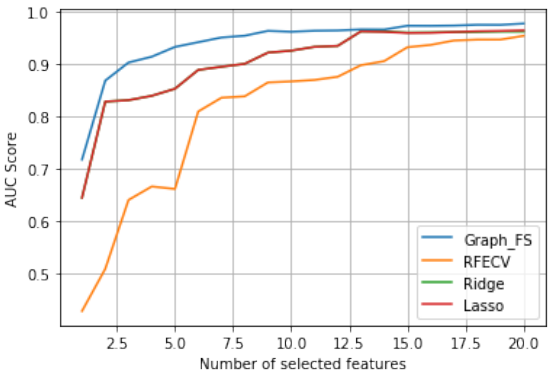
(e) Dataset: caravan



(f) Dataset: madelon



(g) Dataset: Santander



(h) Dataset: spambase

Figure 4.11 – The performance of all feature selection methods in a comparative study. The x-axis is the number of selected features. The y-axis is the performance in terms of AUC. $Graph_{FS}$ shows its effectiveness almost on all datasets except for caravan.

The figure 4.11 presents the performance in terms of AUC score of each feature selection algorithm used in the comparative study. The results show that the Graph-FS drastically outperforms other algorithms (Lasso, Ridge, RFECV) in all datasets (Except for caravan dataset) particularly those that have higher number of features. For example, ds1.100 7.5(g), madelon 4.12(f), santander 4.12(g) and musk 7.5(i) which contain 100, 500, 371 and 167 features respectively.

4.4 Summary of the chapter

In this chapter, we have presented the methods we propose to feature selection aiming to properly select the informative features.

In section 4.2.1, we have described in details the proposed pairwise method in feature selection. The proposed method consists of two main steps. First, it ranks features individually which constitute the filter part. Second, it pair-wisely evaluate features which constitute the wrapper part.

The second method has been presented in the section 4.3.2. The proposed method based on graph representation and on the proposed variable importance criterion. From a created graph using a matrix of individual and pairwise scores, we try to split it into best partitions (communities) then, based on the variable importance criterion, the features that verify the proposed criterion are chosen as the best feature subset.

In future works, we will try to identify the feature that should be used to construct the graph and prevent those with no extra information. This step could speed up remarkably the whole process and it will be easy to execute the proposed algorithm on high dimensional datasets.

The next chapter will be devoted to another proposed FS method which is based on the ensemble technique. The proposed ensemble feature selection method ensures the stability of selected subsets.

5

Ensemble Feature Selection Algorithm

“

“two heads are better than one.”.

John Heywood's.

Chapter 5

Ensemble Feature Selection Algorithm

Contents

Since ensemble feature selection is the recent advance in feature selection literature, in this chapter, we introduce our proposed ensemble feature selection system (EFS). This latter ensures diversity which leads automatically to avoid the redundancy problem in feature selection. Moreover, it ensures the stability of the generated feature subsets. Our proposed approaches are evaluated using eight benchmark datasets. The results show the effectiveness of our ensemble selection approaches.

5.1 Introduction

In this chapter, we propose a new feature selection algorithm based on ensemble selection from libraries of models where each model is trained using one feature only. The main idea of EFS system is to select an optimal ensemble of models from the library that optimizes a given metric. Since each model corresponds to exactly one feature, the subset of selected models corresponds to the subset of optimal features.

First, for each feature we train different models using different classification algorithms and different parameter settings. The set of all models represents our library of models. Second, we use a selection with replacement technique to find the optimal subset of models that when averaged together yield excellent performance.

5.1.1 General Ensemble Selection Framework

The main contribution of this chapter lies in the proposed ensemble feature selection algorithm. We use selection with replacement algorithm to extract and select the well performing ensemble of all models. We add to the ensemble models from the library that maximizes the ensemble's performance on validation set. As we have mentioned, the ensemble of models is selected by maximizing its performance on the validation set. However, its performance is determined by its ability to perform well on unknown datasets (test set). In this situation, overfitting occurs as a serious problem. Moreover, the corresponding performance on the test set does not always increase as indicated in Caruana et al. (2004). This issue leads us to propose the next contributions to improve the ensemble selection.

5.1.2 Ensemble selection improvement

Two solutions are suggested to reduce over-fitting problem and improve ensemble selection.

1. We used Multi-bagging technique introduced by the authors of Caruana et al. (2004). The idea is to select multiple groups (bags) of models randomly from the library. Then, the ensemble selection is done inside each bag. Finally, the returned ensemble is the merging of sub-ensembles returned by each bag. Thanks to this technique, there is no need for validation set.
2. We used a technique called dropout. This technique is a regularization method. It is recently proposed for training deep neuron networks which contain multiple non-linear hidden layers Srivastava et al. (2014). In this contribution, we propose three versions of dropout: Model dropout, instance dropout and Combined dropout. 1) The key idea behind Model dropout is to randomly drop models out from the ensemble during the selection process. 2) In this version we drop out instances instead of models during the selection procedure. 3) Combined dropout is the combination of the model dropout and instances dropout. With dropout, cross-validation and/or splitting data into (train+validation) may be dispensable. Moreover, this technique reduces the co-adaptation between features and instances.

5.2 Related work

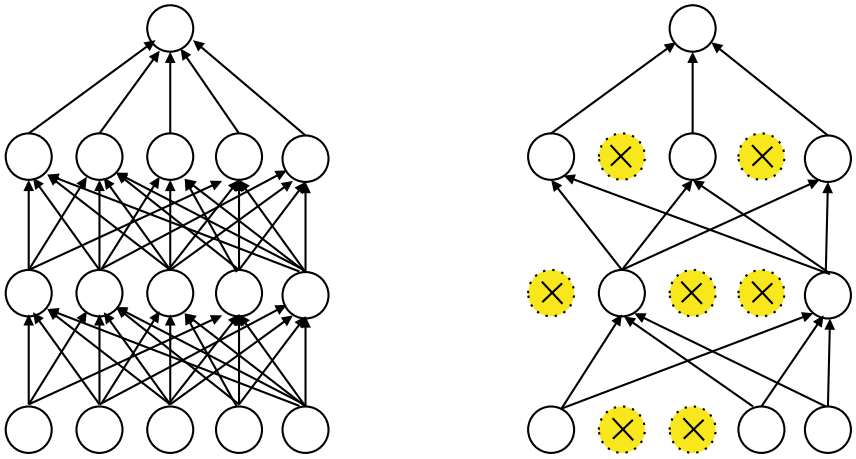
5.2.1 Ensemble selection

Ensemble selection is a popular ensemble learning strategy given its excellent predictive performance concerning various problems. Ensemble selection is a method of constructing ensembles from a library of generated models. Many methods have been proposed to generate diverse and accurate set of models. The

most popular methods are the following: Bagging Breiman (1996) creates many random sub-samples from the original dataset with replacement. This means that we can select the same value multiple times. Then we train models of one type (e.g. Decision tree) using generated sub-samples. Boosting Freund et al. (1999) generates more accurate and diverse set of models, because it starts by predicting original data set and gives equal weight to each observation to force new models so as to attend to those observations that are hard to classify correctly. In Caruana et al. (2004), the authors create a library of 2000 models of each problem using different algorithms and many parameter settings for each algorithm.

5.2.2 Ensemble selection improvement

The main problem that arises when the library contains huge number of models is overfitting. The authors in Caruana et al. (2004) have proposed three additions to the simple selection process (simple forward selection) to overcome this issue. The first addition is selection with replacement, which means models can be chosen several times. The second addition is sorted ensemble initialization, where instead of starting with an empty ensemble, the ensemble is initialized by the N best models. The third addition is bagged ensemble selection, where bags of models are generated randomly from the library, then we select models from those bags. The problem of how much data should be used for the hill-climb set has always been of interest in ensemble selection algorithms. In order to deal with this, the authors of Sun and Pfahringer (2011) have proposed three variations of bagging ensemble selection. In the first variation, the size of data used for hill-climb set is up to the user. In the second variation, the authors used the out of bag instances as hill-climb set. In the third version, single best model is selected at each bagging iteration. The over-fitting problem is a serious issue in deep learning as well. Deep neural nets are a powerful machine learning which contain a huge number of multiple-nonlinear hidden layers and large number of parameters. This makes neural network models learn very complicated relationships between inputs and outputs. Since this results in over-fitting problem, many techniques have been proposed to mitigate its effects. The authors of Srivastava et al. (2014) have proposed what is called dropout technique to prevent over-fitting in neural networks. The units are randomly dropped out from the neural networks during training as shown in figure 5.1 of the paper Srivastava et al. (2014).



(a):Standard Neural Net (b):After applying dropout

Figure 5.1 – Dropout technique. (a) Neural net without applying Dropout. (b) represents Neural net after applying Dropout which forces a neural network to learn more robust features that are useful in combination with many different random subsets of the other neurons.

In the next section, we will describe in details the general ensemble selection framework and the proposed solutions to improve the ensemble feature selection.

5.3 Ensemble Feature Selection

This section is composed of two parts with the first one describing the ensemble selection framework and the second discussing the solutions proposed to improve the ensemble selection.

5.3.1 Ensemble selection framework

Ensemble selection is a method used to construct an ensemble from a library of models. The first step is building a library of models using different classifiers and parameter settings. The second step is the construction procedure of the ensemble. The selection without replacement algorithms such as simple forward selection described in Caruana et al. (2004) selects a well performing ensemble of models. This procedure works as described in the following steps:

1. Start with an empty ensemble.
2. Add to the ensemble the model in the library that maximizes the ensemble's performance on validation set.
3. Repeat Step 2 for a fixed number of iterations or until all the models have been examined.
4. Return the subset of model that yields the best performance on the validation set.

Adding models to the ensemble is done by averaging their predictions with the already selected ones. This kind of algorithms without replacement can select

each model just once. This means that all the selected models in the ensemble have the same weight, which is equal to 1. In other words, the selected models have the same importance. Yet, we should bear in mind that this is not always the case. The simple forward selection is fast and effective but it is sometimes prone to over-fit on the validation set especially when the number of instances is small Kohavi and Sommerfield (1995).

The ensemble's performance using simple forward selection improves as the best models are added to the ensemble. Then it starts dropping when there are just the worst models left in the library. This problem is reduced by using the selection with replacement algorithm where each model can be selected to be in the ensemble more than once.

The selection with replacement technique means that each time we pick a model, we put it back in the library. We prefer using this technique over all models selection without replacement in order to reduce over-fitting problem. In this technique, once the model is selected, its weight increases. The selection process stops depending on its performance on the validation set.

Before starting the selection process, the algorithm initializes the weight of each model at zero. Then, we start the selection process of models, and we add to the ensemble only the models that when averaged together with those in the ensemble maximize the ensemble performance on the validation set. Models can be selected multiple times because each selected model is put back in the library. This step is repeated until no model can produce a notable increase in the ensemble performance. Finally, the weight of each model is returned, and features are sorted according to their weights. The following steps give a more precise description of the algorithm of selection with replacement:

Algorithm 1: Ensemble selection with replacement

Step 1: Ensemble initialization

We initialize the weights of the models at zero.

Step 2: Repeat until all models in the library are used

Add to the ensemble the model thatwhen averaged together with models in the ensemble maximizes the performance ofthe ensemble. Then the weight of the corresponding model is incremented.

Step 3: Return weights

The algorithm output is a dictionary of models and their weight, and the best models are the ones with the highest weight.

Partitioning the available data into (Train+validation) can work well to test the selection with replacement algorithm and deal with the over-fitting problem, but this is not always the case because:

- The performance maximization on validation set does not necessarily mean that the selected ensemble will not over-fit on other datasets.
- The quality of the selected features will decrease especially on datasets with few instances.

5.3.2 Improving ensemble features selection

Two solutions are proposed to reduce over-fitting problem and improve ensemble feature selection performance.

5.3.2.1 Multi-Bagging technique

Over-fitting is a serious problem when the library of models contains a huge number of models. As the number of models in the library increases, the chance to find model combinations that over-fit increases as well. Bagging technique can minimize this issue by creating multiple bags of models. Each bag contains a random sample of models which is less than the number of models in the library. Then inside each bag, we select models from that sample as shown in figure 5.2.

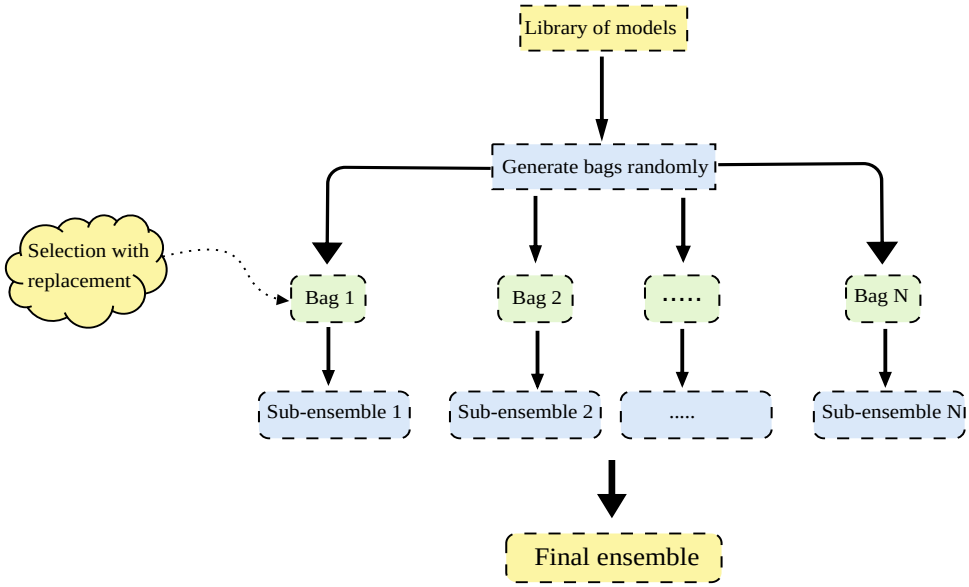


Figure 5.2 – Ensemble feature selection system (EFS). From the initial library, we generate bags at random. We apply selection with replacement on each generated bag so as to select a sub-ensemble of informative features. Finally, by aggregating the generated sub-ensemble together, we construct the final ensemble of useful features as the best subset.

Below is a detailed description of Multi-bagging ensemble selection:

Algorithm 2: Ensemble selection with Multi-bagging

Step 1: Ensemble initialization

- The final ensemble weight is initialized at zero.

Step 2: Repeat until all models in the library are used

- Generate a random bag of models
- Apply the selection with replacement algorithm in order to select the best models from the generated bag.
- The weight of the best models is incremented

Step 3: Return weights

- The final weight is the merging weight of each bag.
- The features whose absolute weight is the greatest are the best features.

5.3.2.2 Dropout technique

Dropout technique is proposed to address the problem of over-fitting in neural network. The idea behind this technique is to randomly drop units from neural network during the training phase. The units have the same probability to be dropped out from the network. By dropping a unit out, we are temporarily removing it from the network with all its incoming and outgoing connections as shown in figure 5.1 In this contribution, we proposed three versions of dropout. The first one is called Model dropout, the second is instances dropout and the third version is Combined dropout.

- **Model dropout** This technique has another way to evaluate ensemble's performance by dropping out models from the ensemble randomly. This prevents models from the co-adaptation (correlation).

  The following steps illustrate how Model dropout works:

Algorithm 3: Model dropout

Step 1: Weight initialization

— All models have the same initial weight equal ZERO.

Step 2: Repeat until no increasing in ensemble performance or all models are examined.

— Drop N models randomly from the ensemble.

— Apply selection with replacement algorithm

Step 3: Return The Ensemble's weights.

— The models are sorted by their weight. The best models are those whose weight is the highest.

- **Instance dropout**
  This method is a variation of dropout where instances are dropped out instead of models (features). At each iteration, an amount of instances is left out of the selection process to minimize the co-adaptation between instances as well, which helps to avoid overfitting problem.
- **Combined dropout**
  This version is a combination of the two versions Model dropout and instances dropout. This means that an amount of instances and features is dropped out at the same time.

5.4 Experimental results and discussion

5.4.1 Model libraries

Model libraries are generated using different classifiers and parameter settings. In our algorithm, each model is trained using just one feature. The library contains (the number of features multiplied by the number of classifiers) models. The final subset of selected features is evaluated using a Random Forest with a grid search strategy for the hyper-parameters. The AUC score is computed using the out of bag (OOB) score of the random forest model.

5.4.2 Conducted Experiments

In this section, we conduct two sets of experiments to evaluate the performance of our proposed ensemble feature selection algorithms. We will first evaluate the ensemble selection with and without replacement technique. We will then demonstrate the applications of the proposed algorithms on eight benchmark datasets.

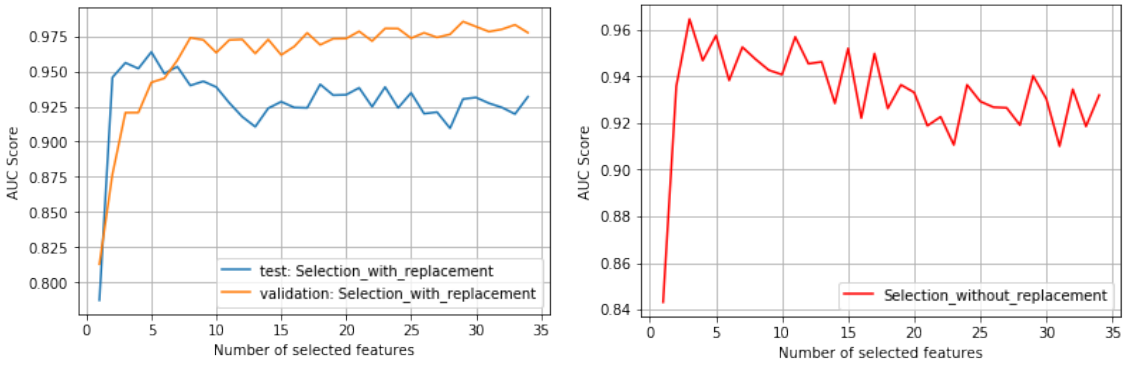
- **Experiment 1: Ensemble feature selection.**
  In the first experiment, we compare feature selection with replacement against feature selection without replacement. This experiment is conducted

to figure out the effectiveness of feature selection with replacement at reducing the problem of overfitting, because this algorithm allows features to be added to the ensemble multiple times. Moreover, the first experiment demonstrates that as the ensemble performance on the validation set increases, the corresponding performance on the test set does not always increase. This means that sometimes the ensemble selection overfits the validation set.

- **Experiment 2: Ensemble feature selection improvement.**
  The second experiment is concerned with evaluating the suggested algorithms which are useful in reducing the problem of over-fitting and improving the ensemble selection performance as demonstrated by the benchmarking results. The algorithms at question here are: Without bagging, Multi-bagging, Model dropout, instances dropout and Combined dropout.

5.4.3 Experiment 1: Ensemble feature selection.

In this experiment, we compare Selection with replacement algorithm to Selection without replacement algorithm described in the introduction section. Moreover, we show how the two algorithms behave on the validation and test set as shown in figure 5.3.



(a) Selection with replacement

(b) Selection without replacement

Figure 5.3 – Selection with and without replacement using validation and test set. (b) represents the selection without replacement where the performance starts declining when the best features are selected. (a) is the selection with replacement. It is clearly illustrated that the selection with replacement can reduce the aforementioned effect but sometimes it over-fits on the validation set (orange curve).

With models selection without replacement, the ensemble performance increases as the best models are added to the ensemble. Once the best models are added, the performance start declining because the best models in the library have been selected and there left just the models that are detrimental to the ensemble. This leads the ensemble to over-fit. This behavior is shown through the red curve of the figure 5.3(b). As we can see after selecting the best 5 features, ensemble selection without replacement starts adding the worst features to the ensemble, which is expressed by the drop in performance. This problem is reduced using models selection with replacement by allowing models to be selected multiple times

as demonstrated by the orange curve 5.3(a). The orange curve is the validation set performance and the blue one is the test set performance. They demonstrate that as the number of features in the library increases, the performance (in terms of AUC) of ensemble selection with replacement on the validation set drastically increases. However, the corresponding performance on the test set does not always increase. This clearly explains that the ensemble features selection sometimes overfits the validation set.

5.4.4 Experiment 2: Ensemble feature selection improvement.

In this experiment, we evaluate our proposed feature selection algorithms and the motives behind the chosen parameters of each algorithm.

1. **Multi-bagging parameters**
   - Number of bags

     The number of bags used is 30. This choice is made after evaluating the algorithm with different numbers of bags as shown in the figure 5.4. We noticed that the more bags used, the better the performance is. In addition, Multi-bagging algorithm becomes more resistant to over-fitting problem, because with more bags the best features have a high chance to be selected over the worst features.

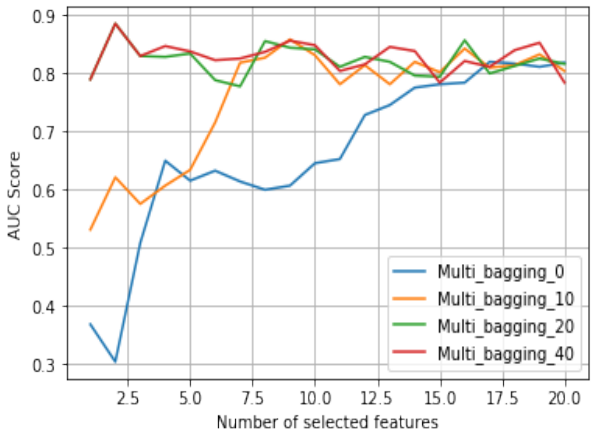


Figure 5.4 – The effect of the number of bags parameter on the performance. The x-axis is the number of selected features. The y-axis is the performance in terms of AUC. It is clear that the more bags we generate, the best performance we achieve. Therefore, the model becomes more resistant to over-fitting issue.

   - Model portion

     This parameter indicates the number of models in each bag. The choice of this parameter is related to the number of bags parameter. We observed from the results conducted that when the number of bags is much higher, it is to choose a large ensemble, and vice versa. In our case, we set Model portion to be one third of the library.

2. **Model dropOut parameters**

- Portion of features dropped out

  This parameter indicates the number of features to be out during the selection procedure at each iteration. This parameter depends on the number of models in the library. The more models in the library the more features should be dropped out from the ensemble.

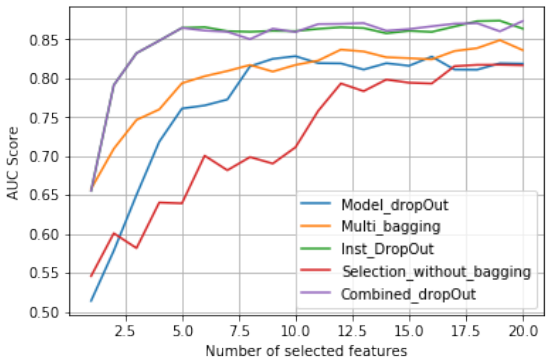
3. **Classifiers used to build library**

   The library of models is generated using decision tree classifier with different depths parameter (max-depth=2, max-depth=3, max-depth=4). Even if the library we generated is not that huge, our algorithms prove to be effective. To reach the best of our proposed algorithms, the library of models should be huge and diverse.

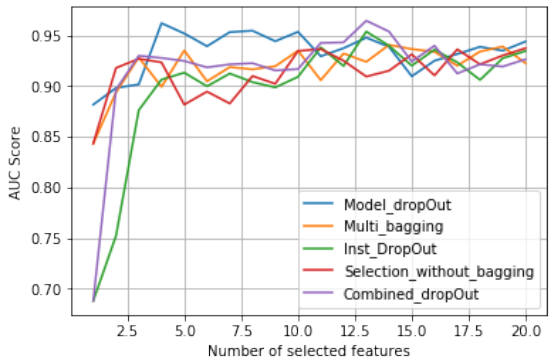
Table 5.1
The best used hyperparameters for each dataset

Dataset	Instance dropped	Model dropped	Number of bags
ds1.100	0.4	0.5	30
credit card clients	0.2	0.6	30
ionosphere	0.1	0.3	30
spambase	0.5	0.3	30
Musk	0.4	0.2	30
Sonar	0.1	0.3	30
numerai	0.4	0.2	30
caravan	0.3	0.3	30

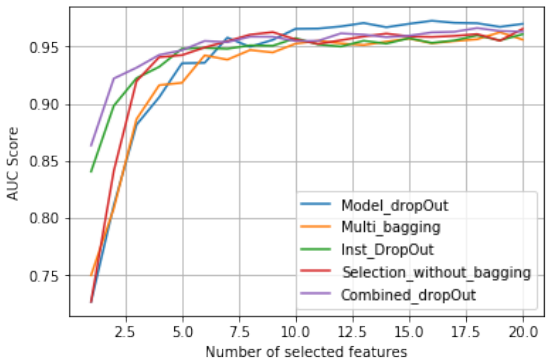
Several observations can be drown from the results of the second benchmark shown in Figure 5.5. First, we found that our Ensemble selection algorithm ($Combined_{dropout}$) outperforms our Multi-bagging, $Instances_{dropout}$ and $Model_{dropout}$ Feature selection algorithms considerably in most of datasets especially datasets with high number of instances such as spambase 5.6(h), ds1.100 7.5(g), credit card client 7.5(d), caravan 7.5(e) and Musk 7.5(i) datasets. Whereas, Multi-bagging and Model-dropOut show their effectiveness on datasets with few number of instances as displayed in datasets sonar 7.5(b) and ionosphere 7.5(a). This is totally logical because $Combiend_{dropout}$ algorithm drops out a portion of instances during the selection process. This reduces the number of instances of datasets. This problem does not appear in datasets with huge number of instances because even if a portion of instances is dropped out during the selection process, the datasets will still have enough instances. This verifies the efficacy of $Compbiend_{dropout}$ and its promising potential for large-scale data mining tasks.



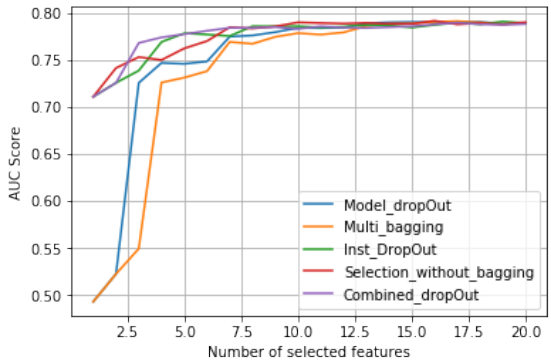
(a) Dataset: ds1.100



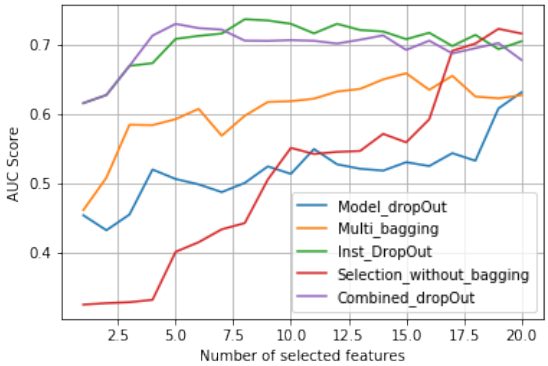
(b) Dataset: ionosphere



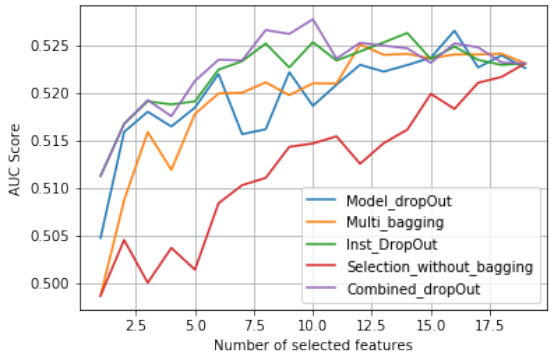
(c) Dataset: Musk



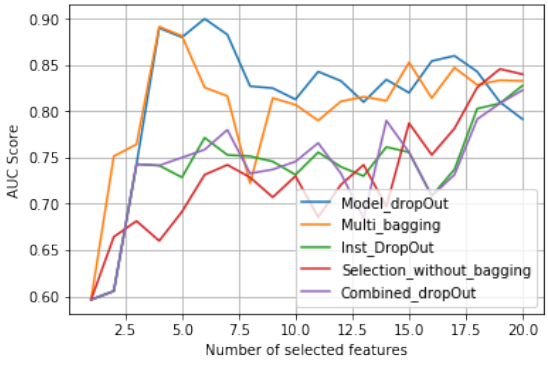
(d) Dataset: credit card



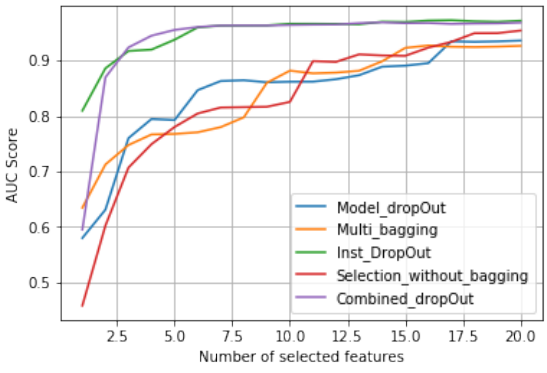
(e) Dataset: caravan



(f) Dataset: numerai



(g) Dataset: sonar



(h) Dataset: spambase

Figure 5.5 – Recorded results on benchmarking datasets using our proposed algorithms. The x-axis is the number of selected features. The y-axis is the performance in terms of AUC. $Model_{dropOut}$, $Multi_{bagging}$, $Inst_{DropOut}$,$Selection_{withoutbagging}$, $Combined_{dropOut}$.

5.5 Summary of the chapter

In this chapter we have introduced a new feature selection approach based on ensemble selection from the library of models, which is an effective learning method over the past several years. The ensemble feature selection algorithms are found to be effective in selecting the best subset among the whole space of features. The main contribution of this paper lies in the proposed ensemble selection framework. First, we created the library of models. Second, we used selection with replacement algorithm to select a well performing ensemble of models. To further improve the proposed feature selection algorithm and reduce the over-fitting problem, we proposed two algorithms, the first uses the multi-bagging technique and the second uses the dropout technique. Experiments with eight benchmarks datasets show that our algorithms work well at selecting the best performing subsets of features.

6

A new noisy random forest based method for feature selection

“

“Machine intelligence is the last invention that humanity will ever need to make.”.

Nick Bostrom.

Chapter 6

A new noisy random forest based method for feature selection

Contents

This chapter presents the method we propose for feature selection using Random Forest importance. the performance of the proposed feature selection method is evaluated using the synthetic and real-world data introduced in **chapter 3**.

6.1 Introduction

Feature selection is an essential pre-processing step in data mining. It aims at identifying the highly predictive feature subset out of a large set of candidate features. Several approaches for feature selection have been proposed in the literature. Random forests (RF) are among the most used machine learning algorithms not just for their excellent prediction accuracy but also for their ability to select informative variables with their associated variable importance measures. Sometimes RF model over-fits on noisy features which lead to choosing the noisy features as the informative variables and eliminating the significant ones. Whereas, eliminating and preventing those noisy features first, the low ranked features may become more important. In this chapter, we propose a new variant of RF that provides unbiased variable selection where a noisy feature trick is used to address this problem. First, we add a noisy feature to a dataset. Second, the noisy feature is used as a stopping criterion. If the noisy feature is selected as the best splitting

feature, then we stop the creation process because at this level, the model starts to over-fit on the noisy features. Finally, the best subset of features is selected out of the best-ranked feature regarding the Gini impurity of this new variant of RF. To test the validity and the effectiveness of the proposed method, we compare it with RF variable importance measure using eleven benchmarking datasets.

This chapter mainly tackled the random forest for feature selection. The major contributions of this chapter are the following:

- Noisy random forest (NRF): We proposed a new variant of RF by adding a new stopping criterion to RF model. First, we added a noisy variable to a dataset; then, during the construction of the tree, if the noisy variable is selected as the best split feature, the construction process should be stopped. This step is meant to ensure the avoidance of the correlated features and the stability of the feature importance.
- Feature ranking: Based on the reliable feature importance of the proposed Noisy random forest, we ranked features in a decreasingly reversed order. This step is reinforced to prevent choosing noisy and un-informative features. Moreover, the elimination is embedded by implication during the training of NRF.
- Feature selection: The explanatory ranked features in feature ranking step are used to construct a sequence of RF models by following a stepwise strategy. Then, the features of the last RF model are selected as the best subset.

Before sinking deep into details, let us put more emphasis on the highly relevant topics of this study (random forest, variable importance, feature selection).

6.1.1 Random Forest

Random forest is a robust algorithm in different applications Breiman (2001). Many researches appreciate RF for their ability to handle the interaction between features, and they can be able to select informative features, especially in expression data analysis Díaz-Uriarte and De Andres (2006a). Based on the aggregation technique Breiman (1996), RF combines several individual classifications or regression trees. Several bootstrap samples are drawn from the training data; then, a set of un-pruned decision trees are constructed on each bootstrap samples, so all trees of the forest are maximal trees. For each tree, a random subset of explanatory variables is selected for each split, and the best split is calculated only within this subset. From the fully constructed forest, the predicted class is obtained as the average of a majority vote of the prediction of all trees in the forest. The estimated prediction error of each tree is obtained using what is called the out of bag samples (OOB), which is a set of observations that is not used for building the trees. Random forest is much stable and accurate as compared to individual trees. Since RF based on ensemble technique, it adjusts the instability that comes from the small changes in the learning sample Strobl et al. (2007). The following steps give a more precise explanation of RF:

1. Create a bootstrapped dataset: we randomly select samples from the original data set.

2. Creating maximum decision trees (without pruning): from the created bootstrapped datasets, we build decision tree using just a random subset of variables at each step.
3. Build a forest by repeating steps 1 and 2 for N times (N decision trees).
4. Predicting the outcome: from the constructed forest, the prediction is obtained as an average or majority vote of the predictions of all trees.
5. Evaluate the model: The prediction error is estimated using the set of observations which are not used for building the current tree (called OOB).

6.1.2 Feature importance

Variable importance (VI) measures of RF have received a lingering momentum in many applied tasks not only at the level of sorting features before a stepwise estimation model, but also in the trend of understanding and interpreting data. The basic variable importance of RF is the mean selection frequencies Strobl et al. (2007). It counts the number of times each feature is selected in all trees. The most selected variable is the most important one. Another widely used VI is the Gini index which measures how well a split on each variable is separating the samples of the two classes in this given node averaged over all trees Strobl et al. (2007); Breiman et al. (1984); Menze et al. (2009). These two indexes are biased and not reliable when features are different in their scales, and when datasets contain many categorical features or noisy ones Strobl et al. (2007). The most advanced variable importance of RF is "Mean decrease accuracy" Strobl et al. (2007); Genuer et al. (2010). This measure is computed when data are permuted in OOB samples: RF importance variable is the difference between the prediction error recorded on out-of-bag samples and the prediction error after permuting the values of data averaged over all trees in the forest. The effect of the scale of measurement and number of categories on mean decrease accuracy is lesser than the effect on the mean selection frequencies and Gini index, but still hugely affects the reliability and interpretability of the variable importance measure.

6.1.3 Feature selection

Various feature selection algorithms based on the variable importance of RF are introduced in the literature. Let us briefly mention some wrapper and embedded method based on variable importance:

1. The first wrapper methods based on VI coming from Classification And Regression Tree method (CART), see Breiman (2001) and of course, random forests Breiman et al. (1984).
2. An algorithm is proposed in Poggi and Tuleau (2006) to select useful variables using a stepwise strategy involving the CART. Based on Support Vector Machine (SVM) scores and relying on descending elimination.
3. The Authors of Rakotomamonjy (2003) proposed a new feature selection method based on Recursive Feature Elimination based Support Vector Machine (SVM-RFE) to evaluate variable subset relevance with a regard to variable selection.

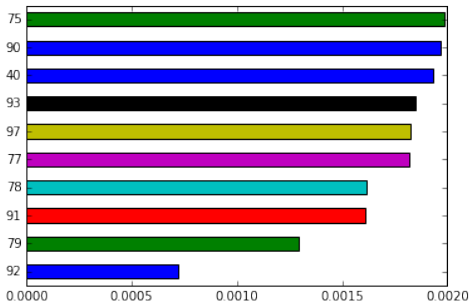
4. Another approach is presented by Díaz-Uriarte and De Andres (2006b). Relying on the Out of bag (OOB-error), it computes variable importance without recalculation at each step as Jiang et al. (2004). Then, after fitting all RF models, the best-chosen solution is the model whose error rate is within U standard error of the minimum error rate of all forests.
5. Two steps algorithm based on random forest importance in proposed in Genuer et al. (2010). In the first step, variables ranked in a decreasing order are meant to identify explanatory variables highly related to the target variable. Then, variables of the smallest importance are to be removed. The chosen variables selected through the first step might be correlated and redundant. The objective of the second step is to select a small number of variables to achieve better accuracy. First, a collection of RF models are constructed using the best variables. The variables leading to the smallest error on OOB samples are selected. Second, a stepwise technique is used to build an ascending sequence of RF. Finally, the variables of the last model are chosen.
6. A guided regularized random forest (GGRF) is proposed in Deng and Runger (2013), where RF model is model on the whole training set then, they utilize the feature importance to guide the feature selection process. In this method, the constructed trees may have high variance. In order to fix the previous problem in GGRF, Deng (2013), proposed a guided random forest (GRF) where each tree in GRF is constructed independently from another.

Two different objectives for variable selection should remain quintessentially fundamental and deep-seated. First, it is of paramount importance to detect the significant features highly related to the response variable. Second, it is highly recommended to select a small subset of variables sufficient to construct an excellent parsimonious prediction of the response variable.

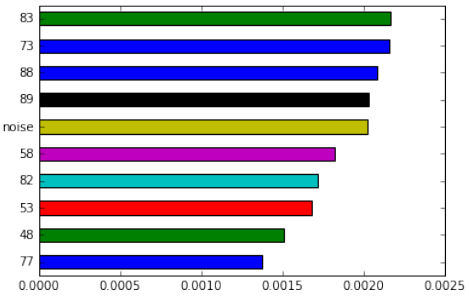
6.2 Motivation

Feature importance measures of random forest are among the widely used criteria as a means of variable selection in many classification tasks. RF importance (Gini impurity, etc.) computes the average decrease in contamination over all trees in the forest due to each feature. Removing the low ranked features according to their importance is not always a practical alternative because the ranking depends on the complexity of the model (tuning parameters). Some low ranked features can be more useful and informative if the complexity of the model is increased and vice versa. To both illustrate and give more details about this behavior, we have conducted a simple experiment using two well-known datasets in the feature selection field: ds1.100 (100 variables) and titanic (26 variables). We have sorted features in decreasing order of RF importance using two scenarios:

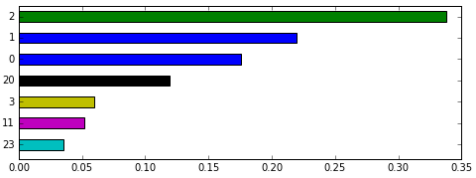
1. The first scenario: We rank the variables in decreasing order using the original features of the two datasets.
2. The second scenario: We rank the variables after adding a generated noisy feature to each dataset. The noisy feature is generated using a normal distribution. This choice based on the conducted experiment in (First experiment in section 4).



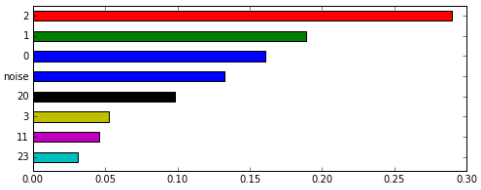
(a) importances without noisy feature , ds1.100(100 attributes)



(b) importances with noisy feature, ds1.100



(c) importances without noisy feature, titanic (26 attributes)



(d) importances with noisy feature, titanic

Figure 6.1 – Ranking features in decreasing order according to RF Gini importance with and without noisy trick. The x-axis represents the features' importance. The y-axis represents features. The feature (noise) is ranked among the best features for both datasets titanic and ds1.100. This confirms that RF model is over-fitting on noisy features.

As it is explicated through the above illustrative Figure 6.1, the noisy feature (noise) is ranked as the best fifth feature among 100 features using the ds1.100 dataset (top right plot). On the titanic dataset, the noisy feature is ranked as the best fourth feature (bottom right plot) although the noisy feature is just a random noise. This is a conclusive empirical research-based assertion that exposes the impracticality of the pre-applied tendency. If we remove the lowest-ranked features first, as it is traditionally implemented by Breiman (2001); Breiman et al. (1984); Genuer et al. (2010), we will probably forget the noisy feature because it is indiscernibly classified among the highly-ranked features. The Random Forest model is probably over-fitting on the noisy features, and by avoiding these noisy features during the moment of constructing the RF model first, some low ranked-features may become more useful to distinguish between classes. Thus, because of the reliable effectiveness of the practical proposed ranking-method mentioned above, there is a possible feasibility of avoiding the selection of noisy features as the most informative ones.

6.3 Proposed method

6.3.1 Procedure

To address the problem of unreliability of RF feature importance (Gini index) discussed in the previous section, we proposed a feature selection method termed " Noisy Random Forest ". The algorithm consists of three main steps.

1. Proposing a new version of random forest called Noisy Random Forest (NRF):

- At each node, a noisy feature is added to the generated subset of features for the sake of splitting the current node.
- The noisy feature is used as a stopping criterion of RF.

2. Feature Ranking:
   - We sort features in decreasing order in accordance with NRF reliable importance.
   - Feature elimination is performed by implication during the selfsame moment of the training phase. All features that are classified below the noisy feature are discarded. Denote by k the remaining features.
   - Feature ranking step allows the selection of more features than necessary in order to make a careful choice later in the next step (feature selection).
3. Feature selection:
   - A stepwise strategy is used to repeatedly construct a sequence of random forest models.
   - Assess the AUC score of the model of the forest at each iteration.
   - Reject a fraction of the least important features.
   - The features of the last model are selected as the best subset.

The following sub-sections provide an in-depth explanation and discussion of each NRF steps.

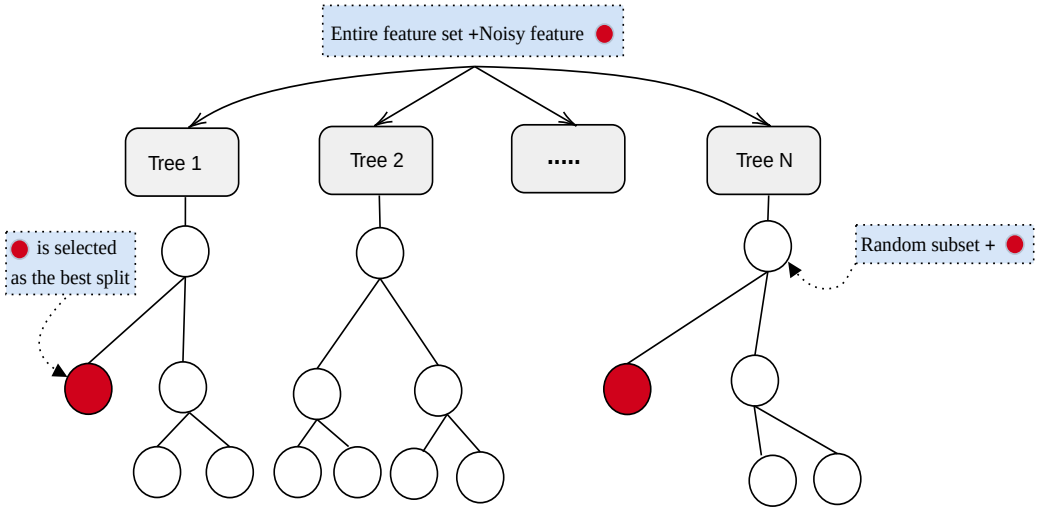


Figure 6.2 – NRF procedure: NRF trees get expanded as RF until the noisy feature is selected as the best split. Then, the splitting process stops in the current branch.

- Why noisy feature is the stopping criterion ?

In RF model, each tree is constructed as the following: at each node, a subset of features S is randomly generated to split the current node. The splitting feature is the one with the maximum information gain (IG). Assuming that the generated

S contains just noisy and useless features, RF will compute IG of all features in S and it will choose the best splitting feature. Then, it will construct the tree without pruning. As a result, the constructed branches are complex and ineffective (of bad quality). This problem has motivated us to propose NRF, where a noisy feature is added to S. Thus, if the noisy (Red circle in Figure 6.2) is selected as the best splitting feature, this means that all features in S are useless and they should not be included in the current tree and the splitting process is stopped. Using NRF, the constructed branches are short and consistent which may lead to construct simple and interpretable trees. In addition, NRF could hugely reduce the computational cost. Instead of computing the IG of all features at each node and constructing the full trees, NRF avoids the construction of sub-trees where the noisy feature is selected as the best split. The complexity of NRF in the worst case would be equal to RF (the case where Noisy feature is not selected which means the splitting process will continue and trees will be fully constructed without pruning as in RF) (see the conducted experiment in table 6.1).

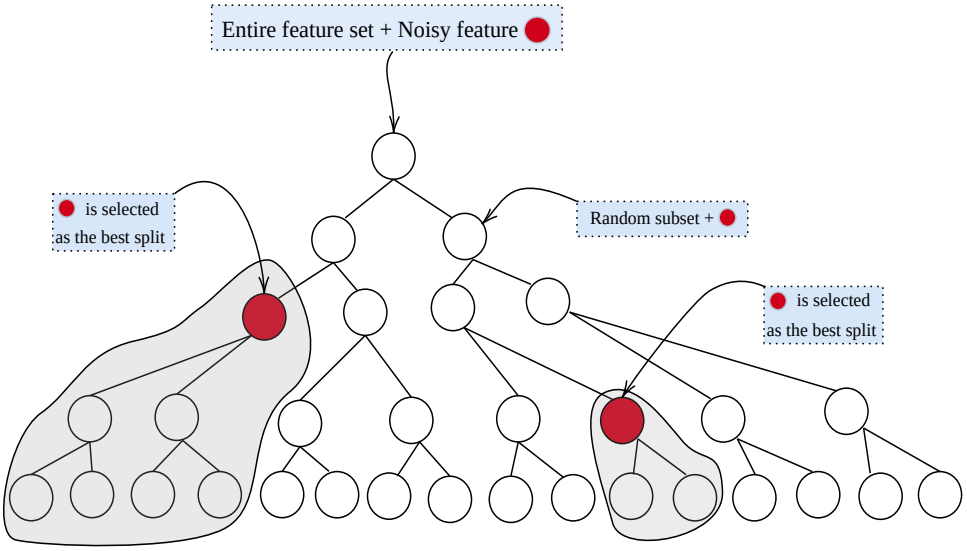


Figure 6.3 – The splitting process in RF and NRF. For RF, trees are fully constructed without pruning. In contrast to RF, NRF constructs a pruned trees using Noisy feature (Red circle). Once the Noisy feature is selected over other candidate features, the splitting process stops.

The modified algorithm of random forest is the following:

Algorithm Noisy random forest (NRF)

1: **Pre-condition:** A training set $S := (x_1; y_1),\ldots,(x_n; y_n)$, features **F**∪{**noisy feature**}, and number of trees in forest **B**
2: **Output:** A forest of trees.
3: **function** CreateTreesForest(**S , F**{**noisy feature**})
4: **H** ←
5: **for i 1,. . . ,B do**
6: hi ← RandomizedTreeLearn(**S , F**{**noisy feature**})
7: H ← H { hi }
8: **end for**
9: **return H**
10: **end function**
11: **function** randomizedTreeLearn**(S, F**∪{**noisy feature**}**)**
12: At each node:
13: **f** ← a subset of **F**
14: **best_split**←choosethe best split feature from **f**∪{**noisy feature**}
15: **If** best_split = noisyfeature **then**
16: **return**the learned tree
17: **Else**
18: split on best_split
19: **return** the learned tree
20: **end function**

6.3.2 Feature ranking

We sort features in decreasing order of NRF importance. As opposed to Genuer et al. (2010); Strobl et al. (2007); Rakotomamonjy (2003); Deng and Runger (2013), our approach does not need variable elimination since it eliminates unimportant and noisy features by implication during the learning process. In the field of feature selection, it is an intelligibly well-known fact that variables with high redundancy might be present in any datasets. Thus, the NRF model can use any of these correlated features. Once one of these correlated features is used as a predictor, the importance of others is exponentially decreased because the impurity, which the correlated features can decrease, is already reduced by the first used feature. Therefore, they will be quantified of below-average and inconsequential importance. As a result of what has been articulated above, the ranked features of the proposed method are not correlated or redundant as is in Genuer et al. (2010); Strobl et al. (2007); Rakotomamonjy (2003); Deng and Runger (2013).

6.3.3 Feature selection

From the subset of the best-ranked features selected in the second step, a necessary attempt should be made to find a small number of features applicable to an excellent parsimonious prediction of the response variable. The stepwise technique is applied when RF models are repeatedly constructed and the worst features are discarded until the examination of all features. The search strategy is guided by the grid search strategy and the AUC score. Thus, the features of the last best performing model are selected.

6.3.4 Complexity analysis

- **Training phase**

  Time complexity is the number of required operations for building models based on data. Time complexity of RF is O(B*m*n*log(n)) where B is the number of constructed trees, m is the number of features to sample at each node and n is the number of data samples Louppe (2014). This is the worst case scenario since RF trees are fully constructed, which means nodes get expanded until all leaves are pure (depth=None). For our NRF model there is always a possibility for nodes to stop expanding, if the "Noise" is selected as the best split (see Algorithm NRF, line 13). This advocates the fact that in the worst cases, the complexity of NRF would be equal to RF complexity; Otherwise, NRF complexity is always less than the one of RF.

- **Selection phase**

  Our suggested method consists of two main steps. The first one is the common feature ranking where features are classified in accordance to their importance. Since our NRF does not allow noisy feature to be included in RF branches, the insignificant features are eliminated by implication during the training phase (as demonstrated in experiment 2 and 3). As a result, the burden of eliminating the un-informative and redundant features is already has been avoided. In the second step of NRF, we put more emphasis on finding a small number of features applicable to an excellent parsimonious prediction of the response variable.

Table 6.1
The execution time (in second) for both RF and our NRF versus the number of times the noisy feature is selected as the best split.

	RF	NRF	#Noisy feature
sonar	30	**29**	7
chess	47.92	**31**	79
spambase	224	**218**	20

To provide more information about the proposed feature selection procedure, the following example is suggested.

6.3.5 Starting example

For further explanation and illustration of the proposed method, we apply the feature selection procedure on the clean dataset, which is a binary classification dataset of 167 attributes and 6600 instances.

- **Training NRF**: We train the new version of the random forest model (NRF), where a noisy feature is used as a stopping criterion. Training shorter trees can be a practical alternative as espoused by the fact that the representative features always appear in the few first levels Poggi and Tuleau (2006). For this reason, the parameters used in NRF are: $Depth = 3$ and number of trees Ntree= 100.

- Feature ranking: After training NRF, features are ranked in a decreasing order according to their NRF importance. All features that are ranked below the noisy one are to be discarded.

The result displayed in the Figure 6.4 shows the ranking of features according to their importance. The high-ranked features are more important than noisy feature. We keep only the features whose importance highly exceed and outperform the noisy feature's poor one. This step leads to retaining more features than necessary. (For the clean dataset, the selected features in the step1 is $k = 19$).

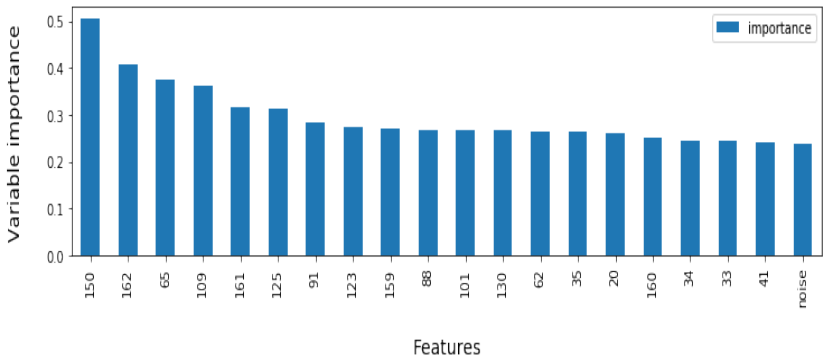


Figure 6.4 – Feature importance using the proposed NRF (clean dataset). The x-axis features. The y-axis is the importance correspond to each feature.

Feature selection: From the K best features, we repeatedly construct a random forest models with a grid search strategy and remove the underperformed feature (lower AUC score). Then, we make a consecutive repetition of this process with the remaining features until all of them are examined in terms of performance. The features of the last model are selected. The following graph Figure 6.5 shows the results of the feature selection step (step2). Note that the AUC score increases quickly and reaches its maximum when the first 17 informative features are included in the model (The AUC score is higher that 96%). Then it remains nearly constant. This means that the best subset contains the first 17 attributes.

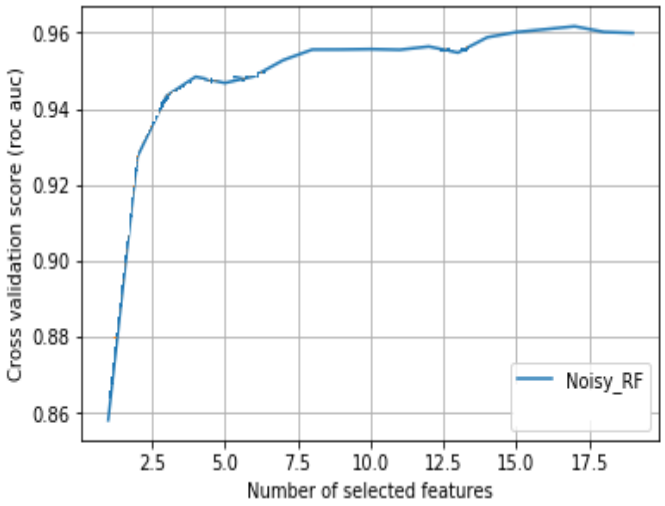


Figure 6.5 – The performance of the feature selection procedure for clean dataset. The x-axis is the number of selected features. The y-axis is the performance in terms of AUC using Cross-validation evaluation technique.

6.4 Experimental results

The proposed method is compared to the standard random forest (RF) approach in terms of prediction AUC score. Three experiments were conducted to assess the validity of our method.

1. **First experiment**: This experiment is conducted to empirically justify and substantiate the choice of normal distribution over other distributions.
2. **Second experiment**: All variables are chosen to be equally irrelevant to scrutinize the stability and the reliability of the proposed method. The reliable variable importance should not prioritize any predictor variable over other.
3. **Third experiment**: This simulation is conducted to evaluate the ability of the proposed algorithm to deal with correlated variables.
4. **Fourth experiment**: This experiment is performed to appraise the performance of the NRF in terms of AUC score using ten standard benchmarking datasets and one high dimensional classification one.

6.4.1 Results and discussion

- **Experiment 1: Why we choose normal distribution**

The used noisy feature in NRF could be generated using different distributions (Normal, geometric, exponential, etc). In the conducted experiments in this manuscript, we choose to use the normal distribution rather than other distributions regarding to its good empirical results in terms of AUC score as illustrated in table 6.2.

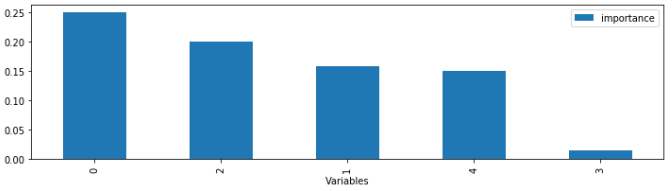
Table 6.2
The impact of different distributions on the performance of NRF in terms of AUC score and execution time(in seconds).

	Normal distribution		Exponential distribution		Geometric distribution		Binomial distribution	
Datasets	Time	AUC	Time	AUC	Time	AUC	Time	AUC
Sonar	30	**92%**	33.15	91.5%	**29.61**	**92%**	31.20	91.3%
chess	47.92	**99%**	44.95	94%	5.10	94%	**4.87**	95%
spambase	224	**97.8%**	206.16	96%	**138.25**	97%	139.67	96.9%
Means	100.64	**96.3%**	94.75	93.8%	**57.65**	94.3%	58.58	94.3%

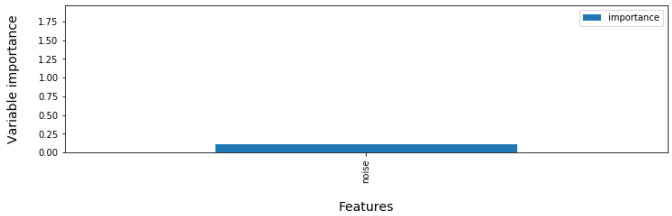
- **Experiment 2: All features are equally irrelevant**

In this simulation, when all features are equally not useful and irrelevant, the variable importance of the random forest (RF) and the proposed NRF (Noisy-RF) are supposed to be the same. However, as it is illustrated and presented in figure 6.6(top plot), the importance of variables is considerably different from one variable to another. As opposed to RF variable importance, the variables' importance of the proposed variant of RF (Noisy-RF) are equally presented and there is no preference of any variable over the others (bottom plot). Accordingly, all of them are deemed to be irrelevant, and therefore they should be discarded. Thus, the drawn conclusion stemmed from the reached implications espouses the following experiment-based assertion: unlike the Gini importance of RF which cannot reliably measure the variable importance, our variable importance measure is dependable and unbiased

Feature selection: From the K best features, we repeatedly construct a random forest models with a grid search strategy and remove the underperformed feature (lower AUC score). Then, we make a consecutive repetition of this process with the remaining features until all of them are examined in terms of performance. The features of the last model are selected. The following graph shows the results of the feature selection step (step2). Note that the AUC score increases quickly and reaches its maximum when the first 17 informative features are included in the model (The AUC score is higher that 96%). Then it remains nearly constant. This means that the best subset contains the first 17 attributes.



(a)



(b)

Figure 6.6 – Variable importance measured by RF and Our NRF. The top plot displays the RF variable importance while the bottom one presents the NRF variable importance. NRF is much better dealing with irrelevant features while RF is not reliably measure the variable importance as it allows irrelevant features to be selected.

- **Experiment 3: The presence of redundant and correlated features.**

To demonstrate the functionality of variable importance's behavior of the standard RF and the proposed variant of RF, an explicatory experiment is conducted in which we applied the proposed method ($Noisy_{RF}$) and Random forest variable importance on a generated dataset that contains correlated variables. The best variable measure is meant to disable the correlated and redundant features from being selected. The results show that the importance measured by the standard RF of all variables are equal (see the top plot) which may allow the redundant and correlated features to be opted as the best feature subset. This problem is tackled through the application of the proposed variable importance measure (bottom plot). Therefore, the selected subset is more consistent and diverse.

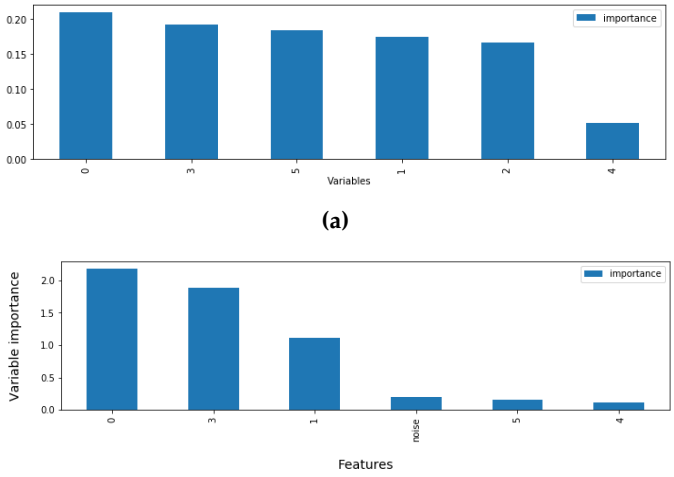


Figure 6.7 – Variable importance measured by RF and Our NRF. (a) The top plot displays the RF variable importance while the bottom (b) one represents the NRF variable importance. NR assigns similar importance to redundant and correlated features which means that those features are allowed to be selected in the final feature subset. This issue is hugely reduced using NRF as it assigns different importance to correlated features to prevent them from being selected.

- **Experiment 4: The performance of Noisy-RF on ten benchmarking datasets.**

In this experimental study, we compared the performance of our feature selection procedure (Noisy-RF) to the Random forest (RF). The comparison is carried out counting on the datasets shown in table 3.1. The quality of the final selected subset for both methods is evaluated through the application of a 3-fold cross-validation to estimate the AUC score rate. So we split each dataset into three stratified folds, each fold is used as a test set, and the remaining folds are used as a training set. The results are obtained as those of Fig. 4, except that for the ranking feature step, we only plot the 50 most important variables to ensure the clarity and the apparent visibility of the graphs. We opted for the usage of the AUC metric because it is more convenient for the evaluation of classifiers performance on unbalanced datasets.

- Standard datasets:

The results obtained for the caravan dataset using our procedure showed that the first step (feature ranking) enables the selection of 26 features only (Figure 6.8(a) the top left plot). After the application of the feature selection step (Figure 6.8(c) the right plot), it is notably conspicuous that with only ten features, the AUC score is in a cumulative growth as it has reached the percentage of 77.2%. The unprecedented attainability of the AUC score remarkably displays the powerful performance of the 10 selected features as opposed to the number of the uninformative features discarded (88.4% of features are eliminated). On the other hand, the results of the Random forest corroborate the fact that despite the large number of

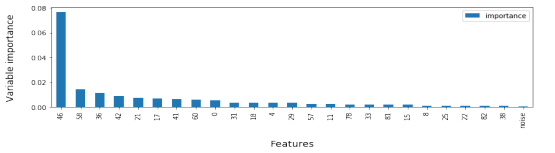
the selected features in the first step, k=47 (Figure 6.8(b)), which provides the potential likelihood of selecting the best features in the second step, the best performance achieved is exclusively restricted in the following percentage of 76.1%.This comparative study confirms that disregarding the powerfully relevant and informative features, the RF variable importance measure tends to select the unreliably biased noisy features as the high-ranked ones.

The results on ionosphere dataset demonstrated that the elimination step (Figure 6.9(a)) prompts the obliteration of 56 % of unimportant features using NRF and the removal of 6% using RF. Then, from the remaining K features, seven features are selected as the reliable subset in step 2 with the maximum AUC score of 97.5% (Figure 6.9(c)) for our NRF. Whereas, the RF maximum performance is restrictively reduced to the following percentage of 97.4%. Thus, no pervasive disparity can be discerned between the maximum AUC score for NRF and RF, yet the amount of features selected by RF in first step is time and memory consuming.

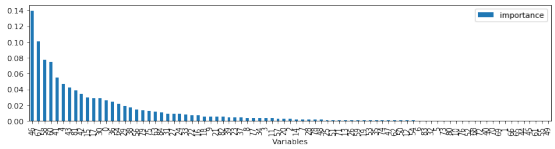
The same NRF elimination procedure when it is applied on spambase dataset leads to the removal of more than 33% of useless features. Out of 66% of the remaining features, 79% are selected from the last model with an AUC score of (97.8%) (Figure 6.10). RF has achieved the same results, yet our proposed method has slightly outperformed it with a further step manifested in the attainability of AUC score and the best-ranked features selection in the first elimination step.

For the clean dataset, which contains 166 features (Figure 6.11), the procedure of feature ranking of NRF engenders the preservation of 19 features only. Whereas, RF leads to the selection of more than 100, which is a largely massive number compared to the pre-selected one of the preserved features. Relying on just the first 17 selected features of our method, the AUC score has reached its maximum. This considerable dimensionality reduction (about 90% of features are eliminated) enables the construction of fast models and decreases the storage and memory requirement.

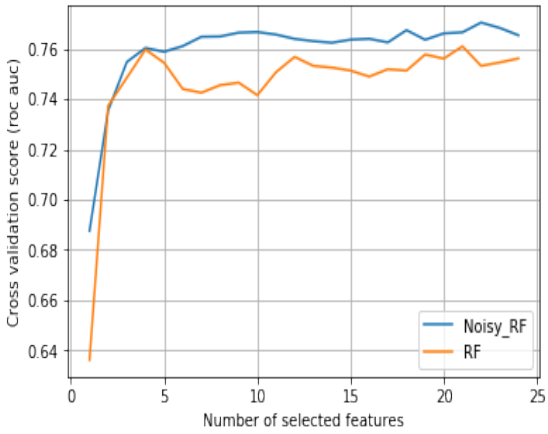
The same results are obtained on datasets eighther, madelon, ticdata2000, credit card, and chess dataset (see the figures Figure 6.12, 6.13, 6.14, 6.15 and 6.16). The elimination step of our proposed method always selects the smallest, reliable and consistent feature subset compared to RF which leads to the achievement of the highest AUC score in the feature selection step. This conducted experiment obviously confirms that features could reliably measure the importance of features by applying NRF even in situation where correlated and redundant features are presented or when features are varied in their scale of measurement. Moreover, the performance of feature subset selected in the feature selection step drastically outperforms RF performance almost in all datasets in terms of AUC score attainability.



(a) NRF importance

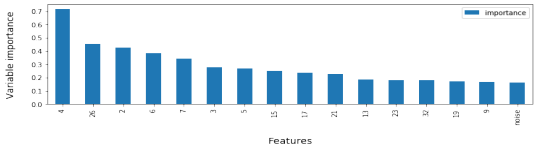


(b) RF imortance

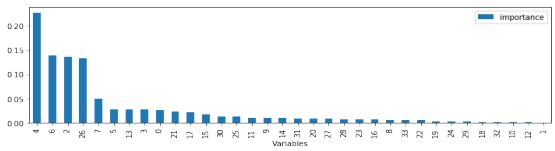


(c) AUC score

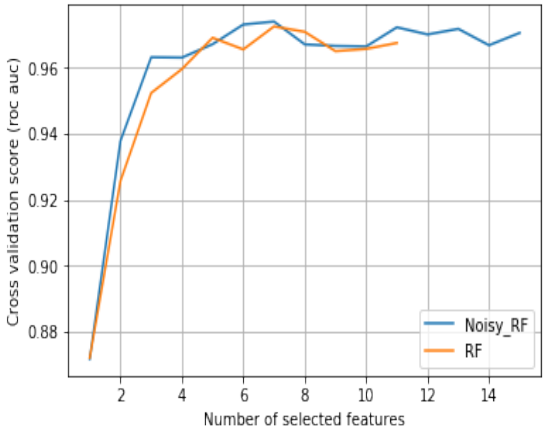
Figure 6.8 – Feature ranking and Feature selection applied on caravan dataset



(a) NRF importance

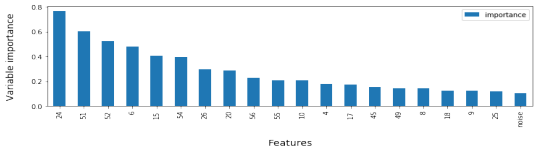


(b) RF imortance

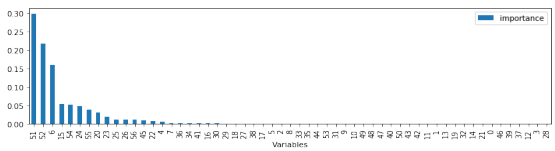


(c) AUC score

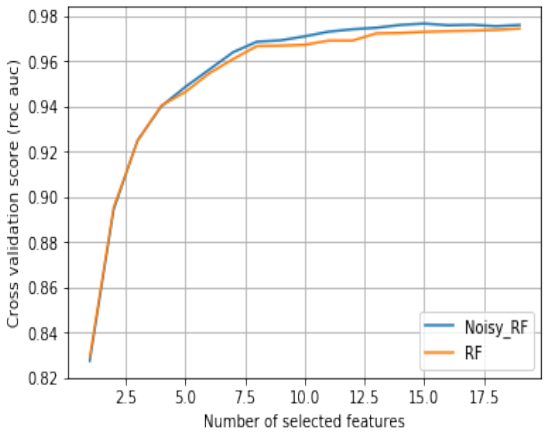
Figure 6.9 – Feature ranking and Feature selection applied on ionosphere dataset



(a) NRF importance

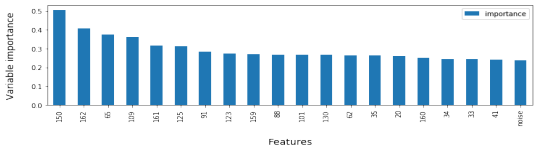


(b) RF imortance

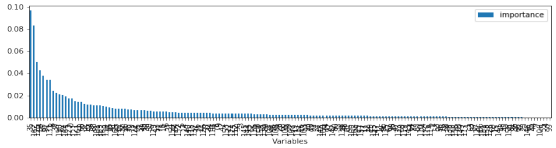


(c) AUC score

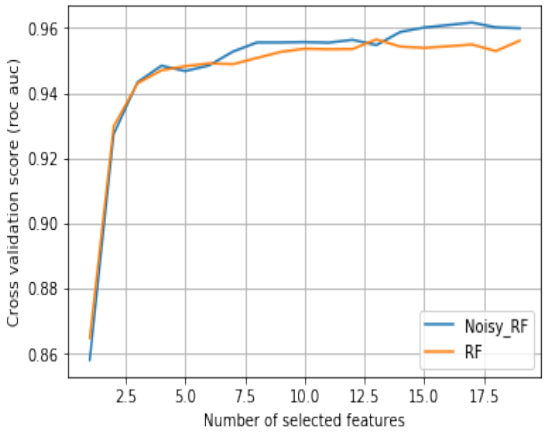
Figure 6.10 – Feature ranking and Feature selection applied on spambase dataset



(a) NRF importance

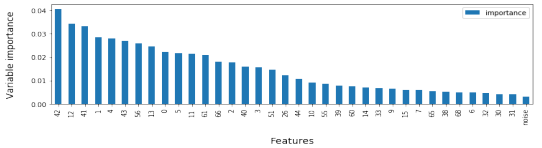


(b) RF imortance

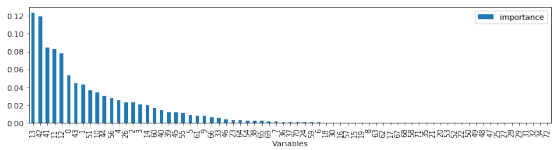


(c) AUC score

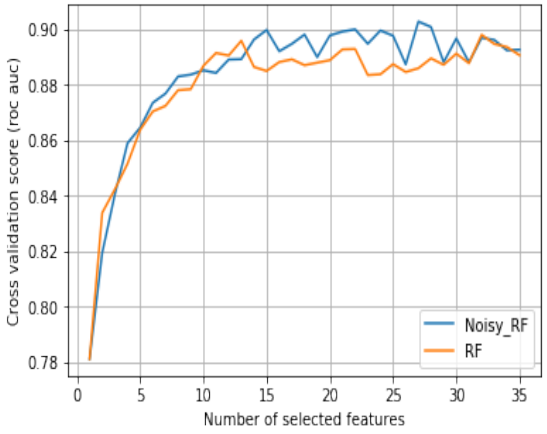
Figure 6.11 – Feature ranking and Feature selection applied for clean dataset



(a) NRF importance

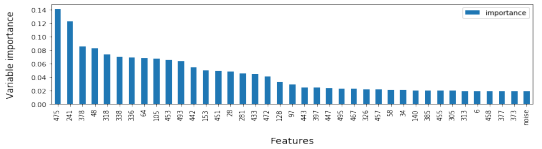


(b) RF imortance

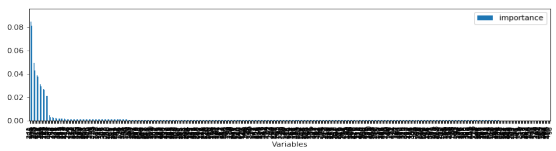


(c) AUC score

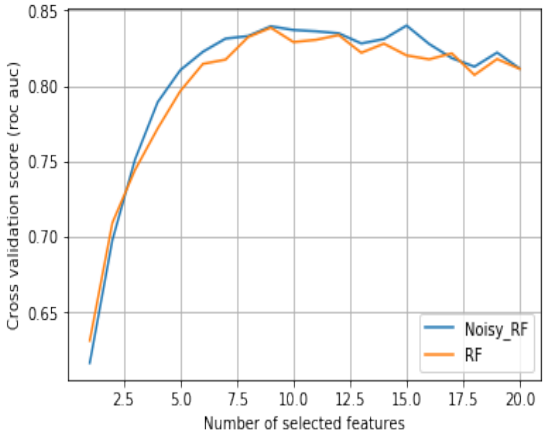
Figure 6.12 – Feature ranking and Feature selection applied for eighther dataset



(a) NRF importance

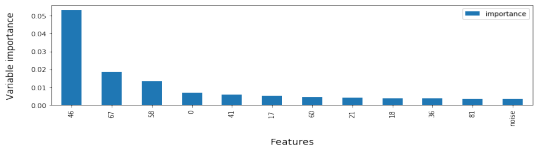


(b) RF imortance

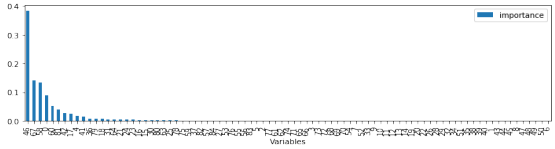


(c) AUC score

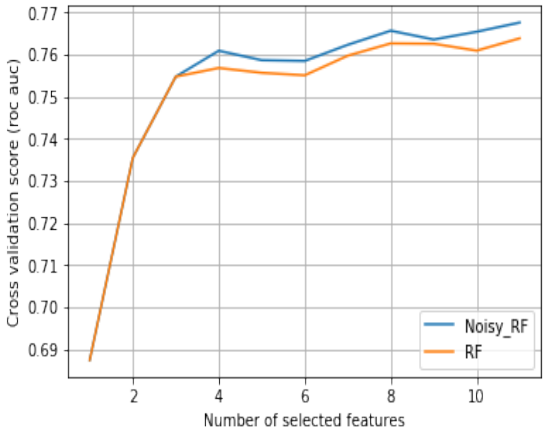
Figure 6.13 – Feature ranking and feature selection for madelon dataset



(a) NRF importance

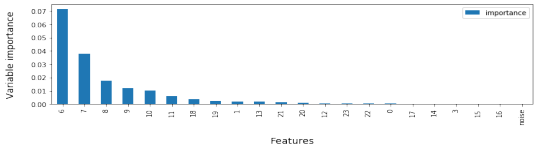


(b) RF imortance

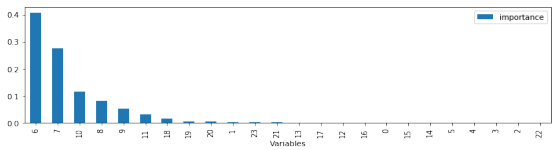


(c) AUC score

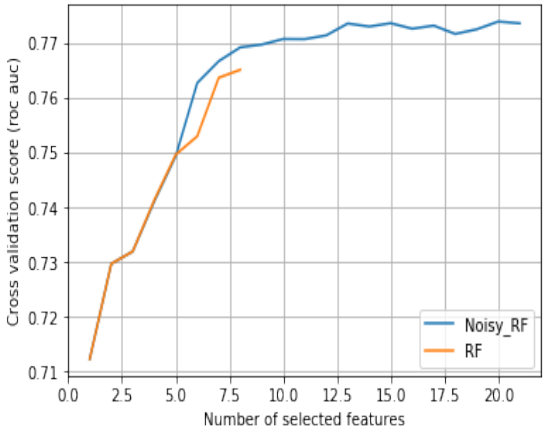
Figure 6.14 – Feature ranking and feature selection for ticdata2000 dataset



(a) NRF importance

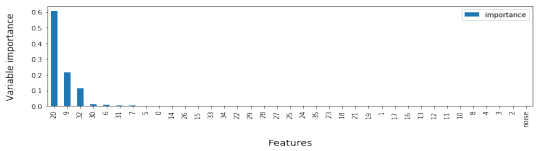


(b) RF imortance

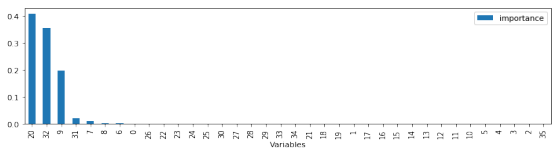


(c) AUC score

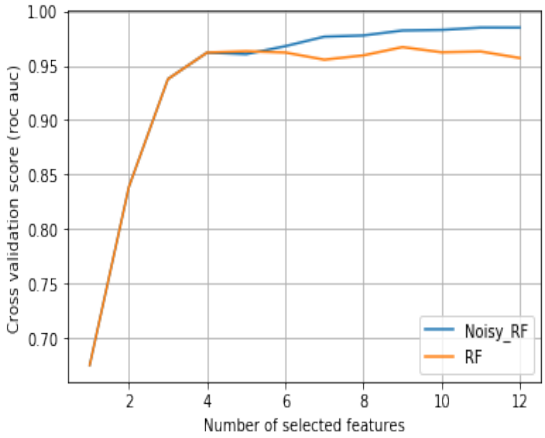
Figure 6.15 – Feature ranking and feature selection for credit card dataset



(a) NRF importance



(b) RF imortance



(c) AUC score

Figure 6.16 – Feature ranking and feature selection for Chess.

- High dimensional classification dataset

The introduced method is also applied on the well-known high dimensional dataset called colon (2000 features and 62 examples) see the table 3.1 to estimate the prediction performance. Since these types of datasets are of small size, we used a 5-fold cross validation so that the training set can contain enough training examples. The drawn results (Figure 6.17) on colon dataset accentuate that the proposed method has the ability to select the best unbiased feature subset even in extreme cases in which datasets contain high number of features or small number of examples.

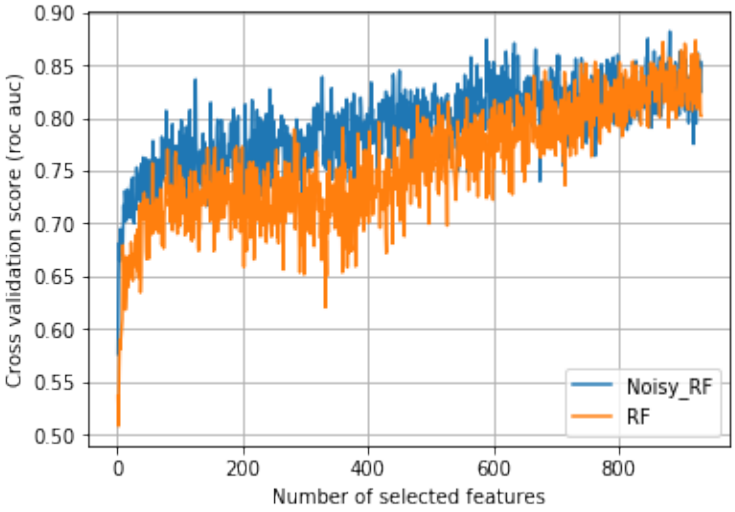


Figure 6.17 – Feature selection for colon(m=62, n=2000) dataset. The x-axis is for the selected features. The y-axis is the cross-validation AUC score. The NRF shows its superiority compared to standard RF.

6.5 Summary of the chapter

Random forest introduced by Leo Breiman in 2001 is a powerful machine learning model that has been applied in many real word problems not just for its outstanding performance but also for its variable importance measures as means of feature selection. Gini impurity is a widely used variable importance measure. Through this chapter, we have demonstrated empirically that this variable importance measure cannot be reliably applied in feature selection in situations when datasets contain huge amount of correlated, redundant and features of varying type and scale of measurement. Therefore, we have proposed an alternative variant of random forest that provides a reliable and unbiased feature importance measure as means of feature selection counting on the proposed noisy feature technique. Our NRF method has shown its ability to reliably measure the variable importance compared to RF. Moreover, it is capable of selecting the smallest consistent and diverse feature subset which leads usually to better performance, minimum resources and storage requirement. In the future works, we will consider the highly advanced variable importance measure which is the mean decrease in accuracy since the effect of the scale of measurement and the number

of irrelevant and correlated features has minor influence. Besides, we will examine and evaluate the new variant of random forest in classification and regression problems.

7

Reinforcement Learning based approach for Feature Selection

“

"A baby learns to crawl, walk and then run. We are in the crawling stage when it comes to applying machine learning."

Dave Waters.

Chapter 7

Reinforcement Learning based approach for Feature Selection

Contents

Beyond the traditional formalization of feature selection problem in the state of the art, in this chapter, we introduce our proposed feedback system based on reinforcement learning approach to perform feature selection. Our proposed system acts as an agent which is able to learn from traversing feature space aiming to select the best feature subset. In order to ensure that the agent keeps exploring the environment, we have proposed a transition similarity measure ***TSM*** in section 7.2.4. At the end of this chapter, additional experiments are conducted so as to compare different proposed feature selection methods in this thesis work against the state-of-the-art will-known methods. The experiments show the superiority and the applicability of our contributions in the field of feature selection when compared with the most popular methods.

7.1 Introduction

Feature selection is the process of identifying the relevant features and removing the irrelevant and redundant ones, intending to obtain the best performing feature subset. This chapter proposes a new feedback feature selection system where a reinforcement learning-based method is used to identify the best feature subset. The proposed system mainly includes three parts. First, decision tree branches are used to traverse the state space to discover new rules and select the best feature subset. Second, a transition similarity measure (TSM) is introduced to ensure that the system keeps exploring the state space by creating diverse branches to overcome the redundancy problem. Finally, the informative features are the most involved ones in constructing the best branches. The performance of the proposed approaches is evaluated on nine standard benchmark datasets. The results using the AUC score show the effectiveness of the proposed system.

In this chapter, we introduce a new feedback system to solve the feature selection problem. The system keeps exploring the state space while it is moving through the available space of features to select the best subset. In this system, we have used the decision tree branches. Therefore, each subset is represented by a branch. The main idea of the proposed feature selection algorithm is to select the best subset of features, which are involved the most at constructing the best branches. In the beginning, the system tries to build the first branch without any prior knowledge (exploring the environment). As iterations take place over and over again, the system gathers experiences that help construct better branches (diverse, relevant, etc.) using the proposed Transition Similarity Measure (TSM). Out of the best branches, we select the most used features in constructing them. The main contributions of this study are fourfold.

1. A reinforcement learning-based method is developed to be used in selecting the best subset of features.
2. The proposed system traverses the state space to select the best subset using a modified version of decision tree branches. Since the transition between states (feature subsets) is controlled using Decision tree branches, the proposed system is simple. As a result, the found solution using our proposed system is easy to interpret.
3. Transition similarity measure (TSM) is intended to ensure that the system keeps exploring the environment by creating new branches and exploiting what it has learned to avoid redundancy and ensure the guaranty of diversity.
4. The proposed system can be adapted to any problem (it is not dependent on a specific dataset) because our feature selection problem is considered as reinforcement learning.

   The remainder of this paper is organized as follows: section two presents the related works. Section three is devoted to the problem and our proposed contributions. In the fourth section, the results of the proposed system are introduced. As to the last section, it is put forward to conclude this work.

Beyond the traditional feature selection (FS) formalization and taking inspiration from the reinforcement learning approach, using our proposed system, the feature

selection problem can be handled in a special way. The feature space using our approach can be seen as a Markov decision process (MDP) Sutton and Barto (2018), where each subset of features is represented by a state (decision tree branch). Our system explores the state space, while it exploits the gathered experiences so far using the proposed transition similarity measure (TSM). In Fard et al. (2013), the authors proposed a method based on reinforcement learning (RL) for selecting the best subset. First, they use an AOR (average of rewards) criterion to identify the effectiveness of a given feature in different conditions. AOR is the average of the difference between two consecutive states in several iterations. Second, they introduce an optimal graph search to reduce the complexity of the problem. The way our system traverses from one state to another is handled using decision tree branches to represent each state, as mentioned before. In its totality, this technique can be considered similar to the way RF creates branches. RF method creates multiple trees. For each tree, only a random subset of input variables is used at each splitting node. Therefore, the final trees of RF are independent of each other, and they lack the concept of learning from the previously created trees. On the other hand, our system adopts this concept. At each iteration, it tries to explore new branches and exploit the gathered knowledge to create better ones in the upcoming iteration.

7.2 The proposed FS system

This chapter introduces a new feature selection system based on reinforcement learning. The proposed system mainly consists of three parts. First, decision tree branches are used to traverse the state space to discover new rules and select the best feature subset. Second, a transition similarity measure (***TSM***) is introduced to ensure that the system keeps exploring the state space by creating new branches and exploiting what it has learned so far to avoid the redundancy problem. Finally, the relevant features are the most involved ones in constructing the branches of quality. For further explanation, the next section will introduce the general framework of reinforcement learning and how our system makes use of the benefits of this powerful approach.

7.2.1 Reinforcement learning problem

RL is the most active and fast-developing area in machine learning and is one of three basic machine learning approaches, alongside supervised learning and unsupervised learning. RL consists of the following concepts: Agent, environment, actions, and reward. The agent takes action A and interacts with an environment to maximize the total reward received R. At iteration t, the agent observes state St from the environment. In return, the agent gets a reward Rt. The agent takes action At. In response, the environment provides the next state S_{t+1} and reward. The process continues until the agent will be able to take the right actions that maximize the total reward. The agent must balance between exploiting what has been learned so far and continuously exploring the environment to gather more information that may help in maximizing the total reward.

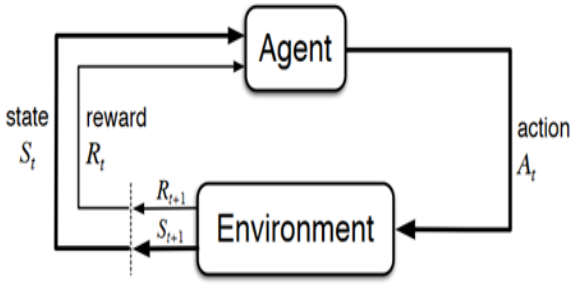


Figure 7.1 – Reinforcement learning framework.

- Agent: An agent takes actions. In our case, the agent is the proposed feature selection system.
- Actions is the ensemble of all possible moves the agent can make, for our system, the actions are the nodes that may be used to create a branch.
- Environment is the feature space through which the system moves. It receives the system's current state and action as input; then, it returns the reward and the next state of the system.
- State is the current situation where the agent finds itself, in our context, is the current node of the branch.

As the reinforcement concepts are represented and explained, the following steps may explain further the main idea of our proposed algorithm. The feature selection system observes the environment and starts with a single node randomly without any prior knowledge (exploration), which branches into possible outcomes. Each of those outcomes leads to the next nodes (action). To indicate how effective the chosen action is, a difference between two consecutive states is produced. While the depth is not reached yet, the system keeps adding one node at a time in order to create a branch. As iterations take place, the system gathers experiences and becomes able to take actions that maximize the overall reward. As a result, branches of high quality are created. A transition similarity measure is proposed to balance between exploiting what it has been learned so far to choose the next action that maximizes rewards, and continuously exploring the feature space to achieve long-term benefits. The way we construct the branch is the same as the decision tree (c4.5), the difference is when we add a node to the branch, we keep just the best branch with the highest threshold. The following steps give more precise information about creating a branch.

7.2.2 Steps to create a branch

We start with a random feature as the root of the branch, then, As long as the branch did not reach the desired depth or min sample leaf yet, the system keeps adding to the branch one node at a time. The added node is the one we obtained using the feature and its threshold that produces the highest AUC score (Area Under the Curve ROC) Ferri et al. (2002). The idea behind using depth and min simple leaf parameters as stopping criteria is to avoid as much as possible the over-fitting problem. The most common stopping method is min sample leaf,

which is the minimum number of samples assigned to each leaf node. If the number is less than a given value, then no further split can be done, and the node is considered as a final leaf node. Besides, the depth of the branch is very useful in controlling over-fitting, because the deeper the branch is, the more information captured by the data and more the splits it has which leads to predict well on the training data. However, it fails to generalize on the unseen data.

Algorithm1: Create a branch	
Step 1:	Create the root node and choose the split feature) Choose the firstfeature randomly.
Step 2:	Compute the best threshold of the chosen feature.
Step 3:	Split the data on this feature into subsets in order to define the node.
Step 4:	Compute the AUC score on left and on right of the node, then, we keep the branch with the best AUC score.
Step 5:	Add the children node to root node.
Step 6:	Choose the next best feature.
Step 7:	Repeat from STEP 2 to STEP 5 until the desired depth or min sample leaf of the branch is reached.

7.2.3 Reward function

A reward function R Fard et al. (2013) is used to calculate the score at each level of the branch by computing the difference between the score of the current branch and its score after a new node is added (DS). The DS indicates how useful the newly added feature is. This function is defined as follows:

$$(AUC_{next} - AUC_{current}) \times log(|Subset_{current}|) \tag{7.1}$$

Where AUC_{next} and $AUC_{current}$ is the score of the current branch and the score after adding a new node, $Subset_{current}$ is the length of samples used to split an internal node.

7.2.4 Transition similarity measure

7.2.4.1 Transition definition

A transition is a process in which something changes from one state to another. In our system, the transition is the link between two successive nodes of the same branch.

7.2.4.2 Transition similarity measure

We proposed a transition similarity measure (TSM) to ensure that our system keeps exploring the state space, learning new rules, and preventing the redundant branches. For each branch, we stock all transitions with the corresponding samples used to split each internal node. Since the algorithm is iterative, different branches may share the same transitions, which is not a problem. In the case when the majority of the samples (higher than a given threshold) are equally used by those transitions of different branches, those two transitions are deemed similar,

which is a huge problem. Because allowing similar transitions to be in different branches can lead to constructing redundant and useless branches. Therefore, the system keeps learning the same rules and branches, This means that the system will be expensive in terms of execution time, while the system should be less resources consuming (run time and storage requirement), and the branches should be strong and diverse. The similarity between two transitions is computed by the following formula:

$$TSM = \frac{|S1 \cap S2|}{|Subset_{current}|} \tag{7.2}$$

Where $|S1 \cap S2|$ is the number of shared samples between two transitions.

7.2.5 Feedback system algorithm

Since the proposed algorithm is iterative, the number of iteration N is given as the input. The reward function is set to zero at the beginning. Our system starts with an empty set F and at each iteration, the system creates a new branch and adds it to F. If the next subset (branch) is already experienced by the system (seen by the system), the system uses this gathered experiences in the upcoming iterations. Otherwise, the system keeps exploring new rules, new patterns, and new branches.

Algorithm 2: Feedback feature selection system pseudo-code

Input:
N: number of iteration
S: Similarity
Output:
R: Reward
for iteration=1 to N **do**:
    F={} to store subsets (branches)
    Step1: Create the root node (**Algorithm 1**)
    Step2: Find all possible transitions (P_t)
    Add the created node to **F** **for** T_i in P_t**do**:
    for T_i in P_t **do**:
        if T_i exist in **F then**:
            Compute the similarity between the two transitions using **TSM**
            if similarity higher than S then:
                f = New node (keep learning and exploring the environment)
                R{F}= $(AUC_{next} - AUC_{current}) \times log(|Subset_{current}|)$
            else:
                $F \cup f$: Add the chosen node to the branch
            end
        end
        Step3: Repeat until the desired depth and min sample leaf is reached
    end
end
Return Reward **R**

7.2.6 An illustrative example

To explain the proposed algorithm further, we suggest the following example. We suppose that we have a dataset of 10 features. The figure bellow (figure 7.2 (a)) contains the whole space of features. The purpose is to select the best subset of features using the proposed system.

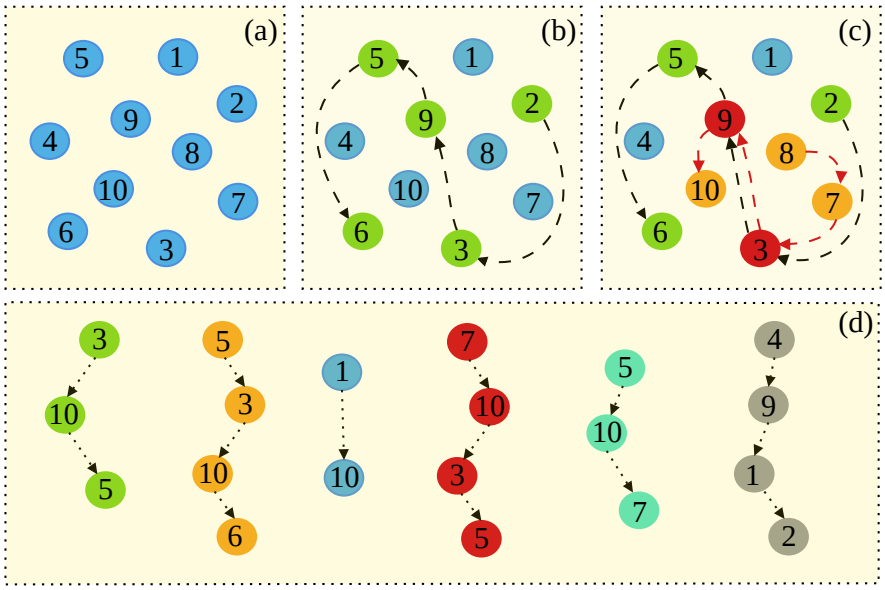


Figure 7.2 – Steps of the FBS proposed algorithm. (a) is the feature space (10 features). (b) is the first created branch (2→3, 3→9, 9 →5, 5→6); randomly, the FBS agent started by the node 2. (c) represents the second created branch where the transition 3→9 is occurred for the second time. At this stage TMS proposed method is involved to prevent the redundant transitions from being included in the created branches. Therefore, the diversity is ensured by implication. (d) shows the final created branches.

- **First iteration 7.2(b)**: The system traverses the features space and creates the first branch without any prior knowledge.

  At each level of the branch, the system stores the AUC score using the reward function R. Moreover, it stores each transition (2→3, 3→9, 9 →5, 5→6) and its corresponding subset of samples.
- **The second iteration 7.2(c)**: As we can see in the second iteration, the transition (3→9) appeared for the second time. Here the TSM (transition similarity measure) should be involved. If two transitions of different branches are similar (nodes with green color), the system should not allow them to be in the next branches (the current branch included). The system has to explore the state's environment to find new rules to prevent the redundancy in creating branches.
- **The N iteration 7.2(d)**: After N iterations, the system is capable of identifying the best branches using the gathered experiences during each iteration. The top ranked branches constructed using the system are the illustrated in the subfigure 7.2(d).

From the above figure 7.2, it is clear that the top subset of features is [10, 5, 3], because those features are involved the most in creating the best branches.

7.3 Experimental results and discussion

This section demonstrates the effectiveness of the proposed feedback system in selecting the best features. Two benchmarks have been conducted, and then the efficiency of our system is evaluated by comparing it with two feature selection (FS) algorithms. The first one is the well-known wrapper algorithm (Recursive Feature Elimination RFE) Wang and Hong (2019). The second one is the pairwise feature selection algorithm, which is recently proposed and proved his effectiveness in identifying the best features Akhiat et al. (2017).

7.3.1 Performance metrics

The quality of the selected feature subset is assessed using the AUC-ROC curve, which is a proper performance measurement for the classification problem. It is a plot of the false positive rate (FPR) versus the true positive rate (TPR) at various thresholds settings between 0 and 1.

7.3.2 Experiments settings

Two benchmarking are conducted to evaluate our proposed system. Firstly, we will demonstrate the applications of the proposed algorithms based on the datasets shown in table 3.1, where FBS is compared with the pairwise method, namely FS-P and with RFE. Secondly, we will demonstrate the ability of the FBS system in finding the best subset as quickly as using only a few features via the second benchmarking. All datasets are divided into two subsets; one subset is employed for training and testing the branches using cross-validation with 3-folds, while the other subset is left out (holdout set) and the performance of the final selected feature subset is evaluated on it. For the sake of a fair comparison, the final selected subset using FBS, FS-P, and RFE is evaluated using a Random Forest with a grid search strategy for the hyper-parameters. The AUC score is computed using the out of bag (OOB) score of the random forest model. Since the benchmarking datasets used in this paper to evaluate the proposed system are unbalanced, the AUC metric is considered the best choice. Moreover, the AUC metric generally can be viewed as a better measure than accuracy Provost (1998).

7.3.3 Feedback system parameters

Feedback system parameters include the value S of the similarity, the value D to indicate the branches depth, and the parameter N is the number of iterations. These parameters are changed from one dataset to another. For example, datasets with large size; the N and D values should be higher since the best branches, in this case, should be more in-depth. The following table 7.1 sums up the best parameters used for each dataset.

Table 7.1
Best parameters for each dataset

Dataset	Iteration N	Depth	Similarity
Credit card	200	7	0.8
Ionosphere	220	4	0.62
Spambase	360	4	0.95
Musk	1000	10	0.65
Sonar	600	3	0.65
Caravan	650	8	0.6
SPECT	100	4	0.7
Eye	110	6	0.9
Numerai	100	5	0.9

As we have mentioned before, the choice of parameters is crucial. The following graph demonstrates the impact of the depth parameter on the quality of the constructed branches using the sonar dataset. We vary this parameter from 1 to 15; then, we plot the training and testing AUC scores.

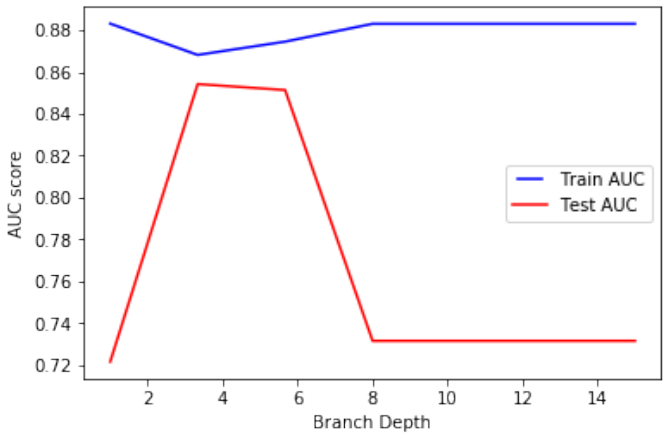


Figure 7.3 – Over-fitting problem. The x-axis is for branch depth. The y-axis is the AUC score. For large depth (larger than 7), the AUC score on training starts increasing while simultaneously the Test AUC starts declining. The best depth that ensures good generalization is for depth=3.

The results on the sonar dataset show precisely that the branches tend to over-fit for large depth values because the branches perfectly predict all of the train data (the blue line), however, they fail to generalize on unseen data (the red line). As can be seen, the best depth for the sonar dataset equals three.

The results on the sonar dataset show precisely that the branches tend to over-fit for large depth values because the branches perfectly predict all of the train data (the blue line), however, they fail to generalize on unseen data (the red line). As can be seen, the best depth for the sonar dataset equals three.

7.3.3.1 First benchmarking

To evaluate our proposed approach FBS, we compare the obtained performance (in terms of AUC score) by FBS with the wrapper method RFE and with the pairwise algorithm FS-P.

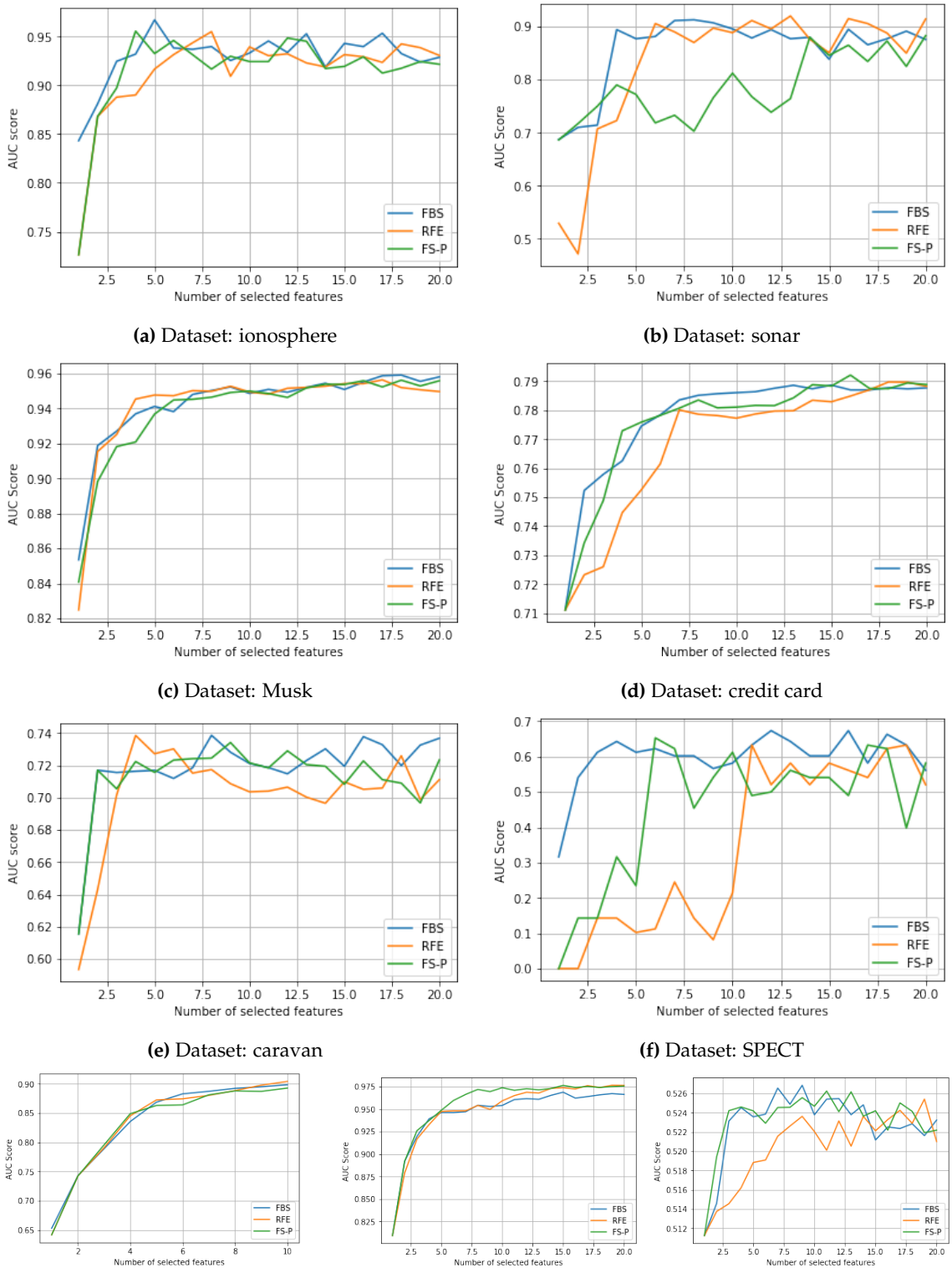
7.3.3.2 Second benchmarking

This benchmarking is conducted to show the ability of the proposed system FBS in achieving the maximum performance using just a few features. For a fair comparison between FBS, FS-P, and RFE, we fix the generated subset size for all algorithms compared as follows: subset of size 5 (FBS_5, $FS-P_5$, RFE_5), a subset of size 10 (FBS_{10}, FS-P_{10},RFE_{10}) and subset of 15 (FBS_{15}, $FS-P_{15}$, RFE_{15}).

7.4 Results and discussion

After selecting the feature subset, the same classifier (RF) is considered to calculate the AUC score. The Random forest is utilized to determine the test performance for the top-ranked features in each used dataset. In Figure 7.4, the results of the comparison between the proposed system performance FBS, pairwise method FS-P, and RFE are presented(first benchmarking). As it is illustrated, our feature selection algorithm FBS outperforms FSP and RFE considerably almost in all datasets such as SPECT (Figure 7.5(f)), credit card (Figure 7.5(d)), ionosphere (Figure 7.5(a)), musk (Figure 7.5(i)), caravan (Figure 7.5(e)), and sonar (Figure 7.5(b)) dataset, except on spambase dataset (5.6(h)). For the numerai dataset (Figure 5.6(f)), our method has a low performance at the beginning compared with others. As our method does not select just the best-ranked feature as a starting point to prevent selecting a suboptimal subset but also tries to maximize the overall performance of the selected subset taking into account the interactions between features. This behavior can be seen after the fifth selected feature on numerai dataset (Figure 5.6(f)), FBS shows its performance drastically over FS-P and RFE.

Table 7.1 shows the best parameters used in our feedback system. The essential insight we can drive from the table is that the choice of the best parameters to use in each dataset is crucial which, means that the parameters should be chosen carefully to construct branches of the best quality.



(a) Dataset: ionosphere

(b) Dataset: sonar

(c) Dataset: Musk

(d) Dataset: credit card

(e) Dataset: caravan

(f) Dataset: SPECT

(g) Dataset: eye

(h) Dataset: spambase

(i) Dataset: numerai

Figure 7.4 – Performance of our system compared with the selected features selected by pairwise method on nine benchmark datasets. The x-axis is for number of selected features for each algorithm. The y-axis is the AUC score performance.

The purpose of the proposed feature selection method is not only to improve the

classification performance but also to yield excellent performance using a minimum number of features (select the number of features as small as possible).

Table 7.2 shows the number of selected features with the highest AUC score on nine benchmarks data sets. As it is illustrated through this benchmarking, FBS selects the proper features compared with FS-P and RFE almost in all datasets. One point to mention here is that the proposed feedback system can find the best subset using a minimum amount of features, as shown in Table 7.2. Thus, the minimum resources requirement, fast execution, and better generalization.

Table 7.2
A comparison of the average classification AUC of FBS, RFE and FS-P based on the first k selected features (k=5, k=10 and k=15)

Dataset	FBS_5	FS-P_5	RFE_5	FBS_10	FS-P_10	RFE_10	FBS_15	FS-P_15	RFE_15
Creditcard	0.775	**0.776**	0.75	**0.786**	0.782	0.779	**0.799**	**0.799**	0.784
Ionosphere	**0.975**	0.951	0.920	**0.930**	0.920	0.931	**0.948**	0.920	0.93
Spambase	**0.950**	**0.950**	**0.950**	0.952	**0.975**	0.960	0.964	**0.975**	**0.975**
Musk	0.940	0.928	**0.949**	**0.950**	**0.950**	**0.950**	0.952	**0.953**	**0.953**
Sonar	**0.89**	0.78	0.81	**0.90**	0.81	0.89	**0.85**	**0.85**	**0.85**
Caravan	0.719	0.719	**0.730**	**0.721**	**0.721**	0.708	**0.720**	0.710	0.710
SPECT	**0.62**	0.25	0.11	0.59	**0.60**	0.20	**0.60**	0.54	0.57
Eye	**0.87**	0.86	0.87	**0.90**	0.89	0.90	-	-	-
Numerai	0.523	**0.524**	0.519	0.524	**0.525**	0.522	**0.60**	0.54	0.57
Mean	**0.81**	0.75	0.70	**0.80**	**0.80**	0.76	**0.80**	0.78	0.79

7.5 Additional Experiments: Contributions vs State-of-the-art FS methods

The goal of this section is to perform an additional experimental study using our previously discussed proposed feature selection algorithms (**FS-P**, **GFS**, **EFS**, **FBS** and **NRF**) and two states of the art feature selection algorithm (**RFE-RF** and **XGBOOST**) so as to scrutinize the efficiency, applicability and adequacy of our methods. The experiments were carried out on 16 datasets of both medium and high dimensionality. The dataset properties (number of features, samples and classes) are listed in Table 3.1. All feature selection methods were executed using the Python tool and 5-fold cross-validation was performed. For the sake of a fair comparison, the quality of the final selected subset of each feature selection method in the comparison is evaluated using a Random Forest with a grid search strategy for the hyper-parameters. The AUC score is computed using the out-of-bag (OOB) score of RF model. The obtained results are reported through the Bar Chart as it is displayed in figures 7.6, 7.7, 7.8 and 7.9. Each figure represents the AUC score attainability of feature selection methods on four datasets. In addition, the summarized results are presented in table 7.4 to enhance the readability and interpretability of the reader.

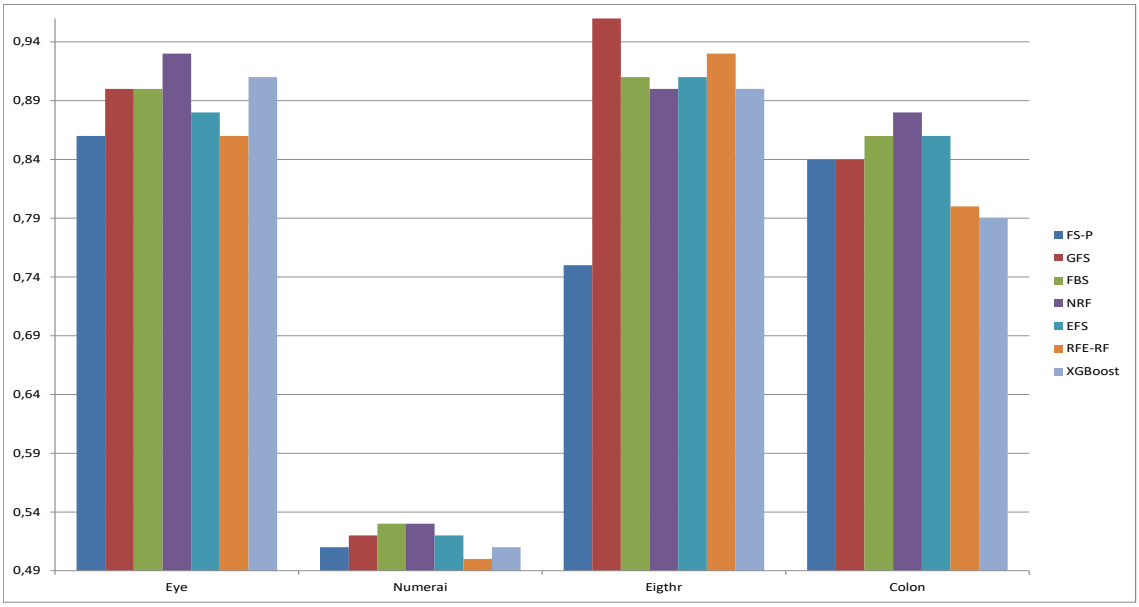


Figure 7.6 – Comparison of our contributions (FS-P, GFS, FBS, NRF and EFS) and state-of-the-art feature selection (RFE-RF and XGBoost). The illustrated results are obtained for ds1.100, Credit card, Chess and Spambase datasets.

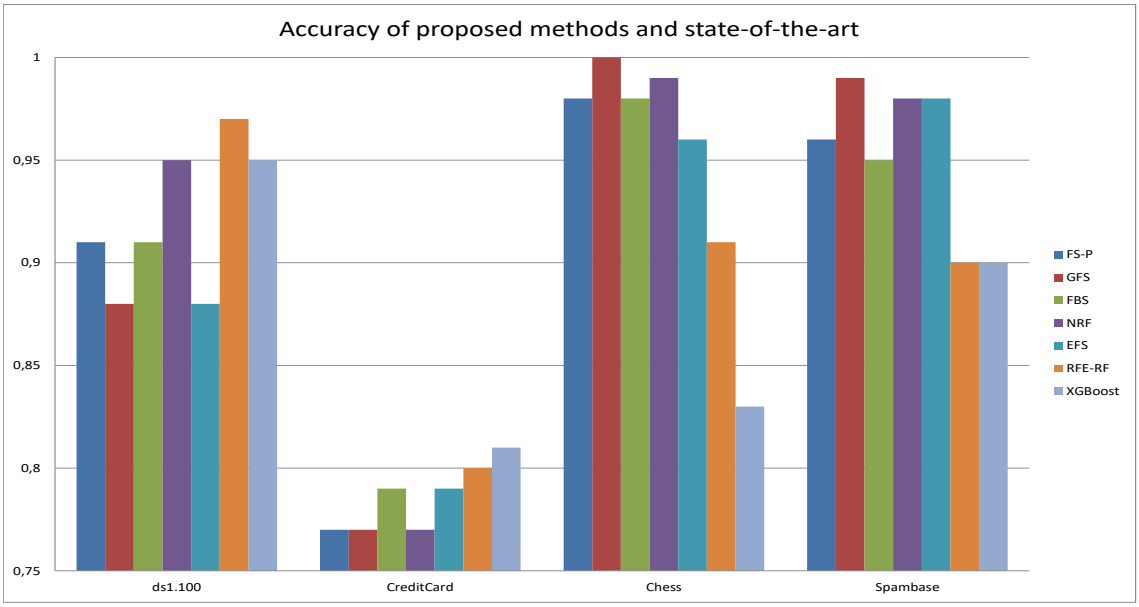


Figure 7.7 – Comparison of our contributions (FS-P, GFS, FBS, NRF and EFS) and state-of-the-art feature selection (RFE-RF and XGBoost). The illustrated results are obtained for Clean, Caravan, Madelon and Santander datasets.

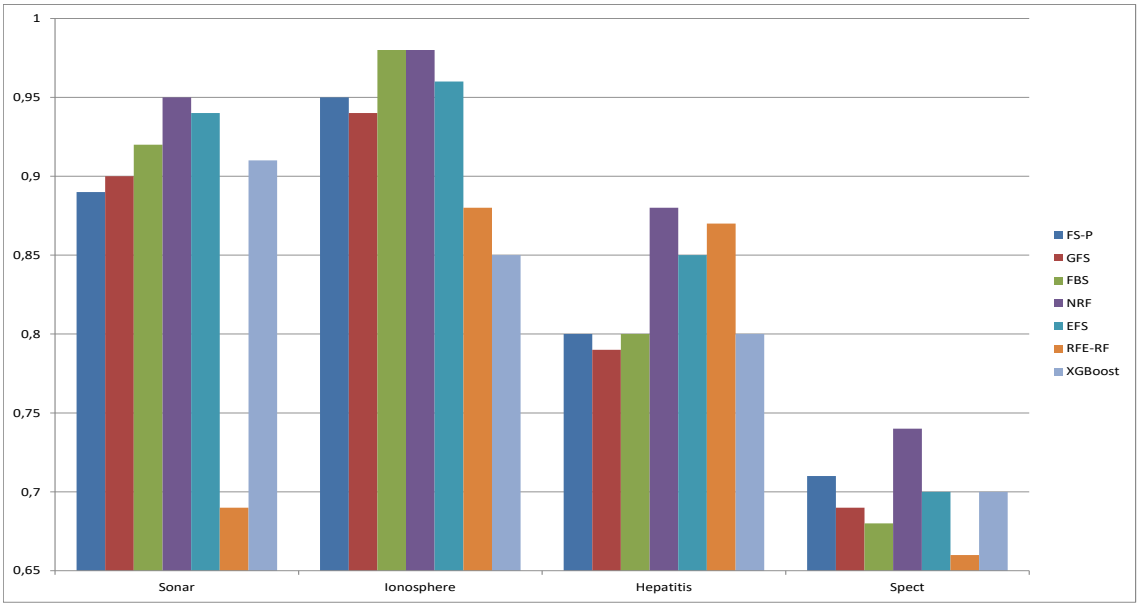


Figure 7.8 – Comparison of our contributions (FS-P, GFS, FBS, NRF and EFS) and state-of-the-art feature selection (RFE-RF and XGBoost). The illustrated results are obtained for Sonar, Ionosphere, Hepatitis and Spect datasets.

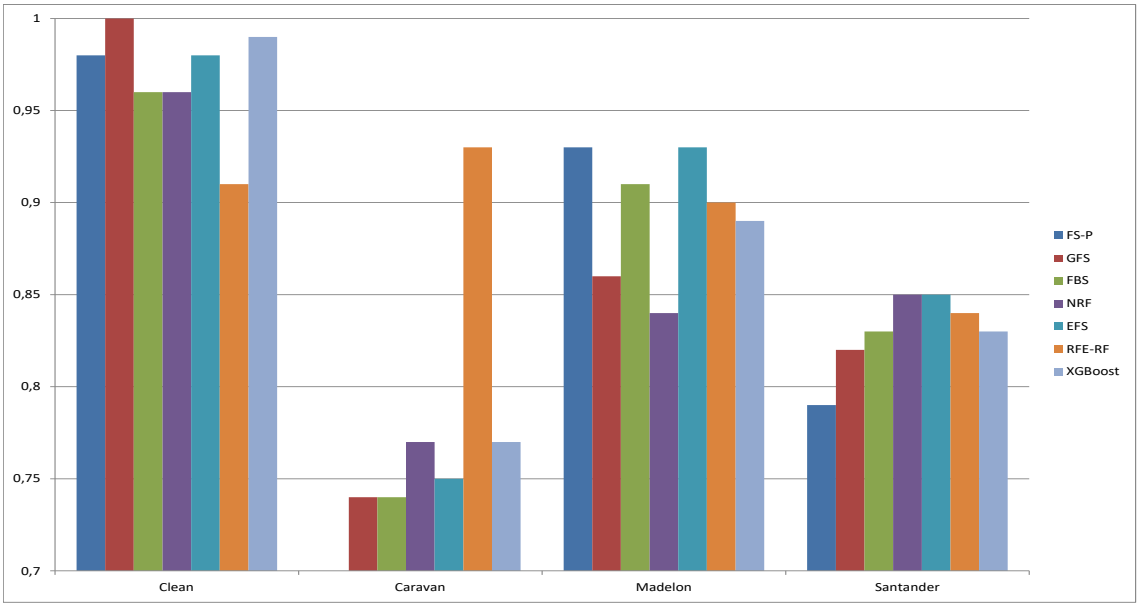


Figure 7.9 – Comparison of our contributions (FS-P, GFS, FBS, NRF and EFS) and state-of-the-art feature selection (RFE-RF and XGBoost). The illustrated results are obtained for Eye, Numerai, Eigthr and Colon datasets.

Table 7.4
The final ranking for our methods and state-of-the-art ones on sixteen datasets. Comparing our methods with each other, it is clear that NRF is ranked as the top ranked one as it leads the ranking in 9 datasets out of 16 followed by EFS, FBS, GFS and FP-S. The overall ranking against RFE-RF and XGBoost has reveals that our contributions can be applied properly achieving good results.

	Proposed Feature Selection Methods					State of the art FS methods	
	FS-P	GFS	EFS	FBS	NRF	RFE-RF	XGBoost
ds1.100	3	4	4	3	2	**1**	2
CreditCard	4	4	3	3	4	2	**1**
Chess	3	**1**	4	3	2	5	6
Spambase	3	**1**	2	4	2	5	5
Clean	3	**1**	3	4	4	5	2
Caravan	5	4	3	4	2	**1**	2
Madelon	**1**	5	**1**	2	5	3	4
Santander	5	4	**1**	3	**1**	2	3
Sonar	4	3	3	2	**1**	5	3
Ionosphere	3	4	2	**1**	**1**	6	5
Hepatitis	4	5	3	4	**1**	2	4
SPECT	2	4	3	5	**1**	6	3
Eye	5	3	4	3	**1**	5	2
Numerai	3	2	2	**1**	**1**	4	3
Eigther	5	**1**	3	3	4	2	4
Colon	6	3	2	2	**1**	4	5

Table 7.3
The summarized results for all tested feature selection methods on sixteen datasets. The number of times each algorithm is ranked as the best one is recorded for readability purposes. The individual ranking shows that NRF performs better than others. The overall ranking cleanly confirms that the proposed methods outperform the state-of-the-art FS methods in the majority of datasets.

	Our contributions					State-of-the-art FS Methods	
Method	FS-P	GFS	EFS	FBS	NRF	RFE-RF	XGBoost
Individual ranking	2\ 16	4\ 16	2\ 16	2\ 16	8\ 16	1\ 16	4\ 16
Overall ranking	11\ 16					5\ 16	

Analyzing these figures 7.6, 7.7, 7.7 and 7.9 and table 7.3 , it is easy to note that the results obtained with our proposed feature selection methods outperformed notably those achieved by the state-of-the-art methods in 11 out of 16 tested datasets. However, in the two datasets where RFE-RF and XGBoost outperforms our methods, our methods have achieved close results (for credit card and ds1.100 dataset).

In table 7.4, a ranking of each feature selection method is shown. Our feature selection NRF seems to be undoubtedly the top ranked method as it is ranked

as the top method in 7 datasets out of 16 and as the top 2 in 4 datasets. These results are followed by those of EFS, GFS and FS-P. For FS-P method, it is ranked as worst method on colon dataset. This is can explain by the fact that FS-P is evaluating features pairwisely only. Since the number of features in colon dataset is very large (n=2000), FS-P cannot detect and identify the underlying interaction that may exist in such space. This problem is can be efficiently using our methods NRF, EFS and FBS as illustrated via the ranking. Overall, the proposed feature selection in this thesis are competitive is terms of AUC and Accuracy score when compared with the state-of-the-art algorithms.

7.6 Summary of the chapter

In this chapter, we proposed a new feature selection system based on decision tree branches concept to represent feature subsets. The proposed system deals with the FS problem as a reinforcement learning problem; the system keeps learning and experiencing new rules with the progress of iterations to build the best branches. After gathering enough knowledge and experiences using the proposed ***TSM***, the proposed system become able to construct the best branches, hence, selecting the best subset of features. To verify the effectiveness of our method, we conducted an extensive set of experiments using nine benchmarks datasets. The results confirm that the proposed feedback system is not only effective at selecting the best performing subsets of features that produce the best performance but also choose the fewest number of features.

Since the proposed feedback system makes use of branches concept to represent the subsets of features, we think that the constructed branches of our system could be used to build a new version of RF. Each tree can learn.

As to the end of this chapter, we have conducted additional experiments. We have compared our contributions with each other and with the well-known feature selection methods (RFE-RF and XGBoost). The recorded results clearly confirmed that the proposed feature selection methods in this thesis have the ability to be applied properly in high dimensional data as they show a competitive performance. On the majority of employed datasets, our proposed methods are ranked from the top 3 best methods.

Conclusion and Future Work

Nowadays, the advent of Big Data has brought unprecedented challenges to machine learning researchers, who now have to deal with the huge amounts of data, in terms of both examples and features, making the learning task more complex and computationally more demanding. Especially, when dealing with an extremely large feature space, learning algorithms' performance can deteriorate due to over-fitting problem, trained models decrease their interpretability as they become more complex, and finally speed and efficiency of the algorithms decline in accordance with size. Thus, feature selection was employed to be able to better model the underlying process of data generation, and to reduce the cost of acquiring the features. Feature selection is a challenging topic in pattern recognition with many potential applications regarding to the continuous growth of data generated from various resources.

The main goal of this thesis was to develop a series of methods in feature selection field to properly select the most informative features/variables and eliminate the redundant and noisy ones. Therefore, enhancing interpretability, reducing complexity, decreasing storage requirement and increasing the performance of the designed models. In addition, a critical analysis and empirical comparisons have been conducted so as to check and expose the advantages and disadvantages of feature selection methods. Moreover, providing to readers and researchers a clear overview and guidelines to properly applying feature selection.

In this section, we present the main conclusions derived from our contributions, and perspectives for future work. In section Summary and Contributions, we present a summary of this work and its resulting contributions. In section Future work we provide the future research directions for expanding and extending our work.

Summary and contributions

We summarize and highlight the main contributions and findings of this thesis work, addressing the research goals described in general Introduction.

Analyzing and evaluating feature selection state of the art methods

Through this thesis, we have critically reviewed and empirically evaluated several feature selection methods from different categories covering phenomena such as the presence of irrelevant and redundant features, the noise in the data or interaction between attributes, including: ***RatlifF, MI*** and ***Chi-2*** as filters, ***SFS, SBS*** and ***RFE-SVM*** as wrappers, ***LASSO, RIDGE*** and ***RF*** selector as embedded. Each method is evaluated in combination with three well-known classifiers to reliably

test their ability to select the best subset. The conducted empirical comparisons are carried out using a large set of benchmarking datasets to illustrate the applicability of feature selection techniques.

In light of the results obtained from the experiments, it is clearly confirmed that too much information does not always help machine learning classifiers since it usually implies that a certain amount of features are redundant or irrelevant, and their presence badly affects the performance of the learning algorithms. Therefore, feature selection pre-processing is a mandatory in order to reduce the data dimensionality, provide insight into the data and enhance the generalization ability.

Another interesting finding that could be derived from the achieved results in subsection (experiment results) is that when dealing with high and complex real-world problems, wrappers cannot be practically applied. Thus, we suggest the use of filters (particularly, ReliefF), since they are independent of the learning algorithm and are much faster than embedded and wrapper methods, as well as having a good generalization ability.

In addition to the presented empirical comparison, we have overviewed the whole feature selection framework so as to supplement readers with a clear overview of the feature selection field. For the stability of feature selection method is an overlooked problem, we have shaded light on the two recent categories ensemble and hybrid since they can yield a consistent and a stable feature subsets. Thus, they could be good alternatives in many machine learning applications.

Developing new feature selection algorithms

Returning to the main goal of this thesis, the second part has been dedicated to proposing novel techniques for large-scale feature selection.

Hybrid filter-wrapper approach: Pairwise Feature Selection

After a detailed review of both featured and recent trends of feature selection modeling, it has been confirmed that wrapper approach can not be applied in high dimensional data. For this reason, recently, numerous studies started to pay more attention to some of the complementary aspects of feature selection such as hybrid approach. In attempt to shed light on this recent trend category, we have suggested a hybrid filter-wrapper method based on evaluating features in pairs. The achieved results showed the superiority of the proposed method especially on small size datasets where the interactions between features are of lower order. In addition the proposed method is beneficial in terms of execution time. By considering the interaction of features (if only in pairs), it is to be expected that pairwise feature ranking performs better than filter methods that evaluate features individually. Whereas the performance could be deteriorate when it has been applied on high dimensional datasets such as madelon (see figure 4.1) which contains 500 features. Another critical issue could arise when considering features individually or in pairs is: Redundancy between features is either not considered at all or only to a limited degree (for pairwise feature selection). Therefore, those redundant features are allowed to be included in the final selected subset.

Graph representation: Graph Feature Selection

The experiments on artificial domains in chapter 4 showed that for pairwise approach is able to detect relevant features under moderate levels of interactions (only on pairs). However, as discussed previously in section 1.6.3, the individually irrelevant features may become weakly or strongly relevant when the interactions with other features are considered, which makes the abandoning interactions between features a critical task. Therefore, a limited "pairwise" approach to detecting feature interactions can not find success in high dimensional datasets where the order of interaction is very large. For this reason, we have proposed a graph feature selection method to capture and modelize as much as possible the underlying interactions between features where each features can interact with all other features. The proposed method has proved its appropriateness dealing with feature interactions as well as its high performance as showed thorough the conducted experiments. The graph feature selection method considerably outperforms two well-known embedded methods (**LASSO**,**RIDGE**) and the state of the art wrapper method **RFECV** on eight benchmarking datasets.

Ensemble Technique: Ensemble Feature Selection

From the perspective of pattern analysis, researchers must focus not only in classification accuracy but also in producing a stable solution. Relying on this fact, we have proposed two feature selection methods based on ensemble technique which is with no doubt the most popular way of making existing feature selection more stable. The underlying idea comes from the field of ensemble learning where different classifiers are combined together to construct an accurate and stable one. In the first method entitled (EFS), we have trained different models using different classification algorithms and different parameter settings to promote diversity of models which then can implicitly enhance the stability of the final subset. In addition, the obtained results showed that the proposed method has also the ability to reduce the over-fitting problem using the bagging technique.

Decision forests are considered the best practice in many machine learning challenges not just for their excellent prediction accuracy but also for their ability to select informative features with their associated variable importance measures (Gini impurity). In chapter 5, we have demonstrated empirically that this variable importance measure cannot be reliably applied in feature selection in situations when datasets contain huge amount of correlated, redundant and features of different type and measurement's scale. Therefore, in the second method, we have proposed an alternative RF so as to produce more stable, reliable and unbiased results. Our proposal has the ability to properly select the most informative features and discard redundant/noisy ones as showed through a plethora of conducted experiments. Moreover, the obtained results on high dimensional datasets have confirmed the applicability of our proposal even in extreme cases such as those of bio-medical tasks where the curse of dimensionality problem can be a serious challenge of many exiting feature selection methods.

Reinforcement Learning for Feature Selection

To handle feature selection problem differently beyond the traditional modelization, we have developed a reinforcement learning based approach for feature selection (FBS). The proposed system acts as an agent that is capable of learning from gathered experiences as it traverses the feature space. To check the efficiency of our method, we conducted an extensive set of experiments using nine benchmarks datasets. The conducted experiment showed a promising results dealing with complex tasks not only at selecting the best performing subsets of features but also at choosing the fewest number of features.

Introducing an evaluation methodology of feature selection

We created grounds for a fair comparison of feature selection algorithms. First, more than fifteen binary classification benchmarking datasets have been discussed and well presented. The datasets are chosen to be different both in terms of the number of instances and attributes to evaluate of the previously discussed algorithms. In order to derive more useful conclusions and insights from the conducted experiments, we have also employed synthetic datasets for the knowledge they provide about the optimal features beforehand. Second, normalization techniques have been introduced to scale all features into a common scale. Third, the entire evaluation methodology developed to check features' discriminatory power is summarized and introduced through the figure 3.1. Four classifiers were built in conjunction with all implemented FS methods are explained in the subsection (3.2.2.1, 3.2.2.2, 3.2.2.3 and 3.2.2.4). Moreover, the validation techniques and evaluation metrics have been used aiming to estimate the generalization performance (Accuracy, AUC, Error estimate, etc) of a model on unseen data.

Future work

There are certainly many possibilities for future research. In this thesis work we open several directions for further perspectives at different levels. In the following, we present the most open questions in feature selection field:

1. Hybrid and embedded feature selection techniques' properties should be explored further.
2. More emphasis should be placed on ensemble feature selection category as it increases the stability of selected feature subsets.
3. Due to the immense volume of generated datasets in many domains including image recognitions, text classification and biomedical, we intend to apply deep learning models to estimates the goodness of features relying on the existed hidden feature interaction that may be well detected using deep learning instead of machine learning models.
4. In the **chapter 4**, a graph feature selection method is proposed where feature space is modelized using a graph representation. The first main step of ***GFS*** consists of constructing a weighted graph using the entire feature space. Here, we suggest adding a pre-processing step where we carefully identify just the features that should be used to construct the graph and prevent the un-informative ones. This extra step could remarkably decrease

the time complexity of ***GFS*** and it could enhance its applicability on high dimensional datasets.

5. In **chapter 6** we have developed a new variant of ***RF*** named ***NRF***. This method showed its effectiveness in selecting the optimal features based on its reliable variable importance. In the future work, we will consider the highly advanced variable importance measure which is the mean decrease in accuracy since the effect of the scale of measurement and the number of irrelevant and correlated features has minor influence. Currently, we are working on examining and evaluating the proposed ***NRF*** as machine learning algorithms in classification and regression problems.
6. We have introduced a reinforcement based approach for feature selection named ***FBS***. This system acts as an agent. It traverses the features' environment trying to maximize the long-term reward by tacking the best actions. At each iteration, the system creates a decision tree branch and from the best ones, it selects the most informative features. As ***FBS*** is an intelligent agent and capable of learning from previously experienced actions, we intend to build an intelligent random forest where each branch (estimator) is created to make use of the already created trees to avoid redundancy, hence, ensuring the diversity.

Publications of the Author

The research works carried out for the completion of this thesis work have resulted in several publications in national conferences, international conferences, book chapter, and specialized journals. We present these publications next, sorting them by publication type.

International journals (#5)

- **Y. Akhiat**, Y. Manzali, M. Chahhou, A. Zinedine, A. (2021). A New Noisy Random Forest Based Method for Feature Selection. Cybernetics and Information Technologies, 21, 10 - 28.https://doi.org/10.2478/cait-2021-0016
- **Y. Akhiat**, M. Chahhou and A. Zinedine, A. (2019). Ensemble feature selection algorithm. International Journal of Intelligent Systems and Applications, 11(1), 24, https://doi.org/10.5815/IJISA.2019.01.03
- **Y. Akhiat**, A. Zinedine, M. Chahhou and. Feature selection methods: Review and empirical study in classification tasks, Journal of Information and Communication Technology (JICT) (**Second Revision**)
- Y. Manzali,**Y. Akhiat**,M. Elmohajir, M. Chahhou, A. Zinedine. Reducing the Number of Trees in a Forest using Noisy Features, Journal of Evolving Systems (EVOS) (**Accepted**)
- **Y. Akhiat**, A. Zinedine and M. Chahhou. Using Reinforcement Learning to Select an Optimal Feature Set. International Journal of SN Applied Sciences(**Under Review**)

International conferences (#6)

- **Y. Akhiat**, Y. Asnaoui, M. Chahhou and A. Zinedine, "A new graph feature selection approach," 2020 6th IEEE Congress on Information Science and Technology (CiSt), 2020, pp. 156-161, https://doi.org/10.1109/CiSt49399.2021.9357067.
- **Y. Akhiat**, M. Chahhou and A. Zinedine, "Feature Selection Based on Graph Representation," 2018 IEEE 5th International Congress on Information Science and Technology (CiSt), 2018, pp. 232-237, https://doi.org/10.1109/CIST.2018.8596467.
- **Y. Akhiat**,M. Chahhou and A. Zinedine, (2017). Feature selection based on pairwise evalution. 2017 Intelligent Systems and Computer Vision (ISCV), 1-6.https://doi.org/10.1109/ISACV.2017.8054919
- **Y. Akhiat**, M. Chahhou, A. Zinedine, and. Reinforcement Learning based approach for Feature Selection, " the 2nd international conference on Embedded Systems and Artificial Intelligence (ESAI'21) (**In Edition by Springer**)

- Y. Bouchlaghem, **Y. Akhiat**, S. Amjad. Feature selection: A review and comparative study, "The seventh International Conference on WIreless Technologies, Embedded and Intelligent Systems (WITS-2021) (**Accepted**)
- Y. Asnaoui, **Y. Akhiat**, A. Zinedine. Feature selection based on attributes clustering, " 2021 Fifth International Conference On Intelligent Computing in Data Sciences (ICDS), 2021, pp. 1-5.0.1109/ICDS53782.2021.9626770

Bibliography

Aha, D.W., Kibler, D., Albert, M.K.. Instance-based learning algorithms. Machine learning 1991;6(1):37–66. URL: https://doi.org/10.1023/A:1022689900470. doi:10.1023/A:1022689900470.

Akachar, E., Ouhbi, B., Frikh, B.. Acsimcd: A 2-phase framework for detecting meaningful communities in dynamic social networks. Future Generation Computer Systems 2021;.

Akhiat, Y., Asnaoui, Y., Chahhou, M., Zinedine, A.. A new graph feature selection approach. In: 2020 6th IEEE Congress on Information Science and Technology (CiSt). IEEE; 2021a. p. 156–161. URL: https://doi.org/10.1109/CiSt49399.2021.9357067. doi:10.1109/CiSt49399.2021.9357067.

Akhiat, Y., Chahhou, M., Zinedine, A.. Feature selection based on pairwise evalution. In: 2017 Intelligent Systems and Computer Vision (ISCV). IEEE; 2017. p. 1–6.

Akhiat, Y., Chahhou, M., Zinedine, A.. Ensemble feature selection algorithm. International Journal of Intelligent Systems and Applications 2019;11(1):24. URL: https://doi.org/10.5815/IJISA.2019.01.03. doi:10.5815/IJISA.2019.01.03.

Akhiat, Y., Chahhou, M., Zinedine, A.. Reinforcement learning based approach for feature selection. In: the 2nd international conference on Embedded Systems and Artificial Intelligence (ESAI'21). Springer; 2021b. .

Akhiat, Y., Manzali, Y., Chahhou, M., Zinedine, A.. A new noisy random forest based method for feature selection. Cybernetics and Information Technologies 2021c;21(2).

Akhtar, F., Li, J., Pei, Y., Xu, Y., Rajput, A., Wang, Q.. Optimal features subset selection for large for gestational age classification using gridsearch based recursive feature elimination with cross-validation scheme. In: International Conference on Frontier Computing. Springer; 2019. p. 63–71. URL: https://doi.org/10.1007/978-981-15-3250-4_8. doi:10.1007/978-981-15-3250-4_8.

Alelyani, S., Tang, J., Liu, H.. Feature selection for clustering: a review. Data clustering: algorithms and applications 2013;29(110-121):144. URL: https://doi.org/10.1109/CIST.2018.8596467. doi:10.1109/CIST.2018.8596467.

Ang, J.C., Mirzal, A., Haron, H., Hamed, H.N.A.. Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection. IEEE/ACM transactions on computational biology and bioinformatics 2015;13(5):971–989. URL: https://doi.org/10.1109/TCBB.2015.2478454. doi:10.1109/TCBB.2015.2478454.

Barshan, E., Ghodsi, A., Azimifar, Z., Jahromi, M.Z.. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. Pattern Recognition 2011;44(7):1357–1371. URL: https://doi.org/10.1016/j.patcog.2010.12.015. doi:10.1016/j.patcog.2010.12.015.

Battiti, R.. Using mutual information for selecting features in supervised neural net learning. IEEE Transactions on neural networks 1994;5(4):537–550.

Belanche, L.A., González, F.F.. Review and evaluation of feature selection algorithms in synthetic problems. arXiv preprint arXiv:11012320 2011;URL: http://arxiv.org/abs/1101.2320.

Blum, A.L., Langley, P.. Selection of relevant features and examples in machine learning. Artificial intelligence 1997;97(1-2):245–271. URL: https://doi.org/10.1016/S0004-3702(97)00063-5. doi:10.1016/S0004-3702(97)00063-5.

Bø, T.H., Jonassen, I.. New feature subset selection procedures for classification of expression profiles. Genome biology 2002;3(4):1–11.

Bolón-Canedo, V., Alonso-Betanzos, A.. Ensembles for feature selection: A review and future trends. Information Fusion 2019;52:1–12.

Bolón-Canedo, V., Sánchez-Maroño, N., Alonso-Betanzos, A.. A review of feature selection methods on synthetic data. Knowledge and information systems 2013;34(3):483–519. URL: https://doi.org/10.1080/02564602.2014.906859. doi:10.1080/02564602.2014.906859.

Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A.. Data classification using an ensemble of filters. Neurocomputing 2014;135:13–20. URL: https://doi.org/10.1016/j.neucom.2013.03.067. doi:10.1016/j.neucom.2013.03.067.

Bolón-Canedo, V., Seth, S., Sánchez-Marono, N., Alonso-Betanzos, A., Príncipe, J.C.. Statistical dependence measure for feature selection in microarray datasets. In: ESANN. Citeseer; 2011. .

Breiman, L.. Bagging predictors. Machine learning 1996;24(2):123–140. URL: https://doi.org/10.1007/BF00058655. doi:10.1007/BF00058655.

Breiman, L.. Random forests; uc berkeley tr567. University of California: Berkeley, CA, USA 1999;.

Breiman, L.. Random forests. Machine learning 2001;45(1):5–32. URL: https://doi.org/10.1023/A:1010933404324. doi:https://doi.org/10.1023/A:1010933404324.

Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.. Classification and regression trees. CRC press, 1984.

Burges, C.J.. A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery 1998;2(2):121–167. URL: https://doi.org/10.1023/A:1009715923555. doi:10.1023/A:1009715923555.

Cai, J., Luo, J., Wang, S., Yang, S.. Feature selection in machine learning: A new perspective. Neurocomputing 2018;300:70–79. URL: https://doi.org/10.1016/j.neucom.2017.11.077. doi:10.1016/j.neucom.2017.11.077.

Canedo, V.B., Marono, N.S.. Novel feature selection methods for high dimensional data. Ph D dissertation 2014;.

Caruana, R., Freitag, D.. Greedy attribute selection. In: Machine Learning Proceedings 1994. Elsevier; 1994. p. 28–36. URL: https://doi.org/10.1016/b978-1-55860-335-6.50012-x. doi:10.1016/b978-1-55860-335-6.50012-x.

Caruana, R., Niculescu-Mizil, A., Crew, G., Ksikes, A.. Ensemble selection from libraries of models. In: Proceedings of the twenty-first international conference on Machine learning. 2004. p. 18. URL: https://doi.org/10.1145/1015330.1015432. doi:10.1145/1015330.1015432.

Chandola, V., Banerjee, A., Kumar, V.. Anomaly detection: A survey. ACM computing surveys (CSUR) 2009;41(3):1–58. URL: https://doi.org/10.1145/1541880.1541882. doi:10.1145/1541880.1541882.

Chandrashekar, G., Sahin, F.. A survey on feature selection methods. Computers and Electrical Engineering 2014;40(1):16–28. URL: https://doi.org/10.1016/j.compeleceng.2013.11.024. doi:10.1016/j.compeleceng.2013.11.024.

Chen, C., Pau, L.. Psp wang handbook of pttern recognition and computer vision world scientific. 1995.

Chen, T., Guestrin, C.. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016. p. 785–794.

Cox, M.A., Cox, T.F.. Multidimensional scaling. In: Handbook of data visualization. Springer; 2008. p. 315–347. URL: https://doi.org/10.1002/wcs.1203. doi:DOI:10.1002/wcs.1203.

Deng, H.. Guided random forest in the rrf package. arXiv preprint arXiv:13060237 2013;URL: http://arxiv.org/abs/1306.0237.

Deng, H., Runger, G.. Gene selection with guided regularized random forest. Pattern Recognition 2013;46(12):3483–3489. URL: https://doi.org/10.1016/j.patcog.2013.05.018. doi:10.1016/j.patcog.2013.05.018.

Díaz-Uriarte, R., De Andres, S.A.. Gene selection and classification of microarray data using random forest. BMC bioinformatics 2006a;7(1):1–13. URL: https://doi.org/10.1186/1471-2105-7-3. doi:10.1186/1471-2105-7-3.

Díaz-Uriarte, R., De Andres, S.A.. Gene selection and classification of microarray data using random forest. BMC bioinformatics 2006b;7(1):1–13. URL: https://doi.org/10.1186/1471-2105-7-3. doi:10.1186/1471-2105-7-3.

Ding, S., Zhu, H., Jia, W., Su, C.. A survey on feature extraction for pattern recognition. Artificial Intelligence Review 2012;37(3):169–180. URL: 10.1007/s10462-011-9225-y. doi:https://doi.org/10.1007/s10462-011-9225-y.

Dreiseitl, S., Osl, M.. Feature selection based on pairwise classification performance. In: International Conference on Computer Aided Systems Theory. Springer; 2009. p. 769–776.

Dunne, K., Cunningham, P., Azuaje, F.. Solutions to instability problems with sequential wrapper-based approaches to feature selection. Journal of Machine Learning Research 2002;:1–22.

Dy, J.G., Brodley, C.E.. Feature selection for unsupervised learning. Journal of machine learning research 2004;5(Aug):845–889. URL: http://jmlr.org/papers/volume5/dy04a/dy04a.pdf. doi:10.1007/springerreference_302701.

Dy, J.G., Brodley, C.E., Kak, A., Broderick, L.S., Aisen, A.M.. Unsupervised feature selection applied to content-based retrieval of lung images. IEEE transactions on pattern analysis and machine intelligence 2003;25(3):373–378. URL: https://doi.org/10.1109/TPAMI.2003.1182100. doi:10.1109/TPAMI.2003.1182100.

Estévez, P.A., Tesmer, M., Perez, C.A., Zurada, J.M.. Normalized mutual information feature selection. IEEE Transactions on neural networks 2009;20(2):189–201. URL: https://doi.org/10.1109/TNN.2008.2005601. doi:10.1109/TNN.2008.2005601.

Fard, S.M.H., Hamzeh, A., Hashemi, S.. Using reinforcement learning to find an optimal set of features. Computers and Mathematics with Applications 2013;66(10):1892–1904. URL: https://doi.org/10.1016/j.camwa.2013.06.031. doi:10.1016/j.camwa.2013.06.031.

Ferreira, A.J., Figueiredo, M.A.. Efficient feature selection filters for high-dimensional data. Pattern recognition letters 2012;33(13):1794–1804. URL: https://doi.org/10.1016/j.patrec.2012.05.019. doi:10.1016/j.patrec.2012.05.019.

Ferri, C., Flach, P., Hernández-Orallo, J.. Learning decision trees using the area under the roc curve. In: Icml. volume 2; 2002. p. 139–146.

Forman, G., et al. An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 2003;3(Mar):1289–1305. URL: http://jmlr.org/papers/v3/forman03a.html.

Freund, Y., Schapire, R., Abe, N.. A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 1999;14(771-780):1612.

Friedman, J., Hastie, T., Tibshirani, R., et al. The elements of statistical learning. volume 1. Springer series in statistics New York, 2001. URL: https://doi.org/10.1007/978-0-387-21606-5. doi:10.1007/978-0-387-21606-5.

Genuer, R., Poggi, J.M., Tuleau, C.. Random forests: some methodological insights. arXiv preprint arXiv:08113619 2008;.

Genuer, R., Poggi, J.M., Tuleau-Malot, C.. Variable selection using random forests. Pattern recognition letters 2010;31(14):2225–2236. URL: https://doi.org/10.1016/j.patrec.2010.03.014. doi:10.1016/j.patrec.2010.03.014.

Ghojogh, B., Samad, M.N., Mashhadi, S.A., Kapoor, T., Ali, W., Karray, F., Crowley, M.. Feature selection and feature extraction in pattern analysis: A literature review. arXiv preprint arXiv:190502845 2019;URL: http://arxiv.org/abs/1905.02845.

Gilad-Bachrach, R., Navot, A., Tishby, N.. Margin based feature selection-theory and algorithms. In: Proceedings of the twenty-first international conference on Machine learning. 2004. p. 43. URL: https://doi.org/10.1145/1015330.1015352. doi:10.1145/1015330.1015352.

Girvan, M., Newman, M.E.. Community structure in social and biological networks. Proceedings of the national academy of sciences 2002;99(12):7821–7826. URL: https://doi.org/10.1073/pnas.122653799. doi:10.1073/pnas.122653799.

Gomez, J.C., Boiy, E., Moens, M.F.. Highly discriminative statistical features for email classification. Knowledge and information systems 2012;31(1):23–53. URL: https://doi.org/10.1007/s10115-011-0403-7. doi:10.1007/s10115-011-0403-7.

Grabczewski, K., Jankowski, N.. Feature selection with decision tree criterion. In: Fifth International Conference on Hybrid Intelligent Systems (HIS'05). IEEE; 2005. p. 6–pp. URL: https://doi.org/10.1109/ICHIS.2005.43. doi:10.1109/ICHIS.2005.43.

Gu, Q., Li, Z., Han, J.. Generalized fisher score for feature selection. arXiv preprint arXiv:12023725 2012;URL: http://arxiv.org/abs/1202.3725.

Guyon, I., Elisseeff, A.. An introduction to variable and feature selection. Journal of machine learning research 2003;3(Mar):1157–1182. URL: http://jmlr.org/papers/v3/guyon03a.html. doi:10.1162/153244303322753616.

Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A.. Feature extraction: foundations and applications. volume 207. Springer, 2008a. URL: https://doi.org/10.1007/978-3-540-35488-8. doi:10.1007/978-3-540-35488-8.

Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L.A.. Feature extraction: foundations and applications. volume 207. Springer, 2008b.

Guyon, I., Weston, J., Barnhill, S., Vapnik, V.. Gene selection for cancer classification using support vector machines. Machine learning 2002;46(1):389–422. URL: https://doi.org/10.1023/A:1012487302797. doi:10.1023/A:1012487302797.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.. The weka data mining software: an update. ACM SIGKDD explorations newsletter 2009;11(1):10–18.

Hall, M.A., Smith, L.A.. Practical feature subset selection for machine learning 1998;.

Hanzo, L., Münster, M., Choi, B., Keller, T.. Ofdm and mc-cdma.

Hapfelmeier, A., Ulm, K.. A new variable selection approach using random forests. Computational Statistics & Data Analysis 2013;60:50–69. URL: https://doi.org/10.1016/j.csda.2012.09.020. doi:10.1016/j.csda.2012.09.020.

Haury, A.C., Gestraud, P., Vert, J.P.. The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. PloS one 2011;6(12):e28210.

Hindawi, M.. Feature selection for semi-supervised data analysis in decisional information systems. Ph.D. thesis; INSA de Lyon; 2013.

Hosmer Jr, D.W., Lemeshow, S., Sturdivant, R.X.. Applied logistic regression. volume 398. John Wiley & Sons, 2013. URL: https://doi.org/10.1002/0471722146. doi:10.1002/0471722146.

Hossin, M., Sulaiman, M.. A review on evaluation metrics for data classification evaluations. International Journal of Data Mining and Knowledge Management Process 2015;5(2):1. URL: https://doi.org/10.5121/IJDKP.2015.5201. doi:10.5121/IJDKP.2015.5201.

Hsu, H.H., Hsieh, C.W., Lu, M.D.. Hybrid feature selection by combining filters and wrappers. Expert Systems with Applications 2011;38(7):8144–8150. URL: https://doi.org/10.1016/j.eswa.2010.12.156. doi:10.1016/j.eswa.2010.12.156.

Huggins, J.H., Rudin, C.. A statistical learning theory framework for supervised pattern discovery. In: Proceedings of the 2014 SIAM International Conference on Data Mining. SIAM; 2014. p. 506–514. URL: https://doi.org/10.1137/1.9781611973440.58. doi:10.1137/1.9781611973440.58.

Jain, A., Zongker, D.. Feature selection: Evaluation, application, and small sample performance. IEEE transactions on pattern analysis and machine intelligence 1997a;19(2):153–158. URL: https://doi.org/10.1109/34.574797. doi:10.1109/34.574797.

Jain, A., Zongker, D.. Feature selection: Evaluation, application, and small sample performance. IEEE transactions on pattern analysis and machine intelligence 1997b;19(2):153–158. URL: https://doi.org/10.1109/34.574797. doi:10.1109/34.574797.

Jain, A.K., Chandrasekaran, B.. 39 dimensionality and sample size considerations in pattern recognition practice. Handbook of statistics 1982;2:835–855.

Jiang, H., Deng, Y., Chen, H.S., Tao, L., Sha, Q., Chen, J., Tsai, C.J., Zhang, S.. Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. BMC bioinformatics 2004;5(1):1–12. URL: https://doi.org/10.1186/1471-2105-5-81. doi:10.1186/1471-2105-5-81.

Jiang, S.y., Wang, L.x.. Efficient feature selection based on correlation measure between continuous and discrete features. Information Processing Letters 2016;116(2):203–215.

Jin, X., Xu, A., Bie, R., Guo, P.. Machine learning techniques and chi-square feature selection for cancer classification using sage gene expression profiles. In: International Workshop on Data Mining for Biomedical Applications. Springer; 2006. p. 106–115. URL: https://doi.org/10.1007/11691730_11. doi:10.1007/11691730_11.

John, G.H., Kohavi, R., Pfleger, K.. Irrelevant features and the subset selection problem. In: Machine Learning Proceedings 1994. Elsevier; 1994. p. 121–129. URL: https://doi.org/10.1016/b978-1-55860-335-6.50023-4. doi:10.1016/b978-1-55860-335-6.50023-4.

Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.. An efficient k-means clustering algorithm: Analysis and implementation. IEEE transactions on pattern analysis and machine intelligence 2002;24(7):881–892. URL: https://doi.org/10.1109/TPAMI.2002.1017616. doi:10.1109/TPAMI.2002.1017616.

Khalid, S., Khalil, T., Nasreen, S.. A survey of feature selection and feature extraction techniques in machine learning. In: 2014 science and information conference. IEEE; 2014. p. 372–378. doi:10.1109/SAI.2014.6918213.

Kittler, J.. Feature set search algorithms. Pattern recognition and signal processing 1978;.

Kohavi, R., John, G.H.. Wrappers for feature subset selection. Artificial intelligence 1997;97(1-2):273–324. URL: https://doi.org/10.1016/S0004-3702(97)00043-X. doi:10.1016/S0004-3702(97)00043-X.

Kohavi, R., Sommerfield, D.. Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. In: KDD. 1995. p. 192–197. URL: http://www.aaai.org/Library/KDD/1995/kdd95-049.php.

Koller, D., Sahami, M.. Toward optimal feature selection. Technical Report; Stanford InfoLab; 1996.

Kononenko, I.. Estimating attributes: Analysis and extensions of relief. In: European conference on machine learning. Springer; 1994. p. 171–182. URL: https://doi.org/10.1007/3-540-57868-4_57. doi:10.1007/3-540-57868-4_57.

Krızek, P.. Feature selection: stability, algorithms, and evaluation. Ph.D. thesis; Ph. D. thesis, Czech Technical University in Prague; 2008.

Kudo, M., Sklansky, J.. Comparison of algorithms that select features for pattern classifiers. Pattern recognition 2000;33(1):25–41. URL: https://doi.org/10.1016/S0031-3203(99)00041-2. doi:10.1016/S0031-3203(99)00041-2.

Kuncheva, L.I.. A stability index for feature selection. In: Artificial intelligence and applications. 2007. p. 421–427.

Kyburg Jr, H.E.. Probabilistic reasoning in intelligent systems: Networks of plausible inference. 1991. URL: https://doi.org/10.1016/0004-3702(91)90084-W. doi:10.1016/0004-3702(91)90084-W.

Lavrač, N., Gamberger, D., Blockeel, H., Todorovski, L.. Knowledge Discovery in Databases: PKDD 2003: 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003, Proceedings. volume 2838. Springer, 2003. URL: https://doi.org/10.1007/b13634. doi:10.1007/b13634.

Li, F., Yang, Y.. Analysis of recursive feature elimination methods. In: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. 2005. p. 633–634. URL: https://doi.org/10.1145/1076034.1076164. doi:10.1145/1076034.1076164.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H.. Feature selection: A data perspective. ACM Computing Surveys (CSUR) 2017;50(6):1–45. URL: http://arxiv.org/abs/1601.07996. doi:10.30534/ijatcse/2020/50922020.

Likas, A., Vlassis, N., Verbeek, J.J.. The global k-means clustering algorithm. Pattern recognition 2003;36(2):451–461. URL: https://doi.org/10.1109/IJCNN.2008.4634069. doi:10.1109/IJCNN.2008.4634069.

Liu, H., Motoda, H.. Computational methods of feature selection. CRC Press, 2007. URL: https://doi.org/10.1016/j.ipm.2009.03.003. doi:10.1016/j.ipm.2009.03.003.

Liu, H., Motoda, H., Yu, L.. A selective sampling approach to active feature selection. Artificial Intelligence 2004;159(1-2):49–74. URL: https://doi.org/10.1016/j.artint.2004.05.009. doi:10.1016/j.artint.2004.05.009.

Louppe, G.. Understanding random forests: From theory to practice. arXiv preprint arXiv:14077502 2014;.

Maglogiannis, I.G.. Emerging artificial intelligence applications in computer engineering: real word ai systems with applications in ehealth, hci, information retrieval and pervasive technologies. volume 160. Ios Press, 2007.

Mandal, M., Mukhopadhyay, A.. An improved minimum redundancy maximum relevance approach for feature selection in gene expression data. Procedia Technology 2013;10:20–27. URL: https://doi.org/10.1016/J.PROTCY.2013.12.332. doi:10.1016/J.PROTCY.2013.12.332.

Mangal, A., Holm, E.A.. A comparative study of feature selection methods for stress hotspot classification in materials. Integrating Materials and Manufacturing Innovation 2018;7(3):87–95. URL: http://arxiv.org/abs/1804.09604. doi:10.1007/s40192-018-0109-8.

Matlab, R.. Version 8.1. 0.604 (r2013a). Natrick, Massachusetts: The MathWorks Inc 2013;.

McHugh, M.L.. The chi-square test of independence. Biochemia medica 2013;23(2):143–149. URL: https://doi.org/10.11613/BM.2013.018. doi:10.11613/BM.2013.018.

Menze, B.H., Kelm, B.M., Masuch, R., Himmelreich, U., Bachert, P., Petrich, W., Hamprecht, F.A.. A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data. BMC bioinformatics 2009;10(1):1–16. URL: https://doi.org/10.1186/1471-2105-10-213. doi:10.1186/1471-2105-10-213.

Miao, J., Niu, L.. A survey on feature selection. Procedia Computer Science 2016;91:919–926. URL: https://doi.org/10.1016/j.procs.2016.07.111. doi:DOI:10.1016/J.PROCS.2016.07.111.

Minasny, B.. The elements of statistical learning, trevor hastie, robert tishirani, jerome friedman, (2009), springer series in statistics, isbn 0172-7397, 745 pp. 2009.

Mitra, P., Murthy, C., Pal, S.K.. Unsupervised feature selection using feature similarity. IEEE transactions on pattern analysis and machine intelligence 2002;24(3):301–312. URL: https://dblp.org/rec/journals/pami/X02. doi:10.1109/34.990133.

Molina, L.C., Belanche, L., Nebot, À.. Feature selection algorithms: A survey and experimental evaluation. In: 2002 IEEE International Conference on Data Mining, 2002. Proceedings. IEEE; 2002a. p. 306–313. URL: https://doi.org/10.1109/ICDM.2002.1183917. doi:10.1109/ICDM.2002.1183917.

Molina, L.C., Belanche, L., Nebot, À.. Feature selection algorithms: A survey and experimental evaluation. In: 2002 IEEE International Conference on Data Mining, 2002. Proceedings. IEEE; 2002b. p. 306–313. URL: https://doi.org/10.1109/ICDM.2002.1183917. doi:10.1109/ICDM.2002.1183917.

Narendra, P.M., Fukunaga, K.. A branch and bound algorithm for feature subset selection. IEEE Computer Architecture Letters 1977;26(09):917–922. URL: https://doi.org/10.1109/TC.1977.1674939. doi:10.1109/TC.1977.1674939.

Naseriparsa, M., Bidgoli, A.M., Varaee, T.. A hybrid feature selection method to improve performance of a group of classification algorithms. arXiv preprint arXiv:14032372 2014;URL: http://arxiv.org/abs/1403.2372. doi:https://doi.org/10.5120/12065-8172.

Neumann, J., Schnörr, C., Steidl, G.. Svm-based feature selection by direct objective minimisation. In: Joint Pattern Recognition Symposium. Springer; 2004. p. 212–219. URL: https://doi.org/10.1007/978-3-540-28649-3_26. doi:10.1007/978-3-540-28649-3_26.

Newman, M.E.. The structure and function of complex networks. SIAM review 2003;45(2):167–256. URL: https://doi.org/10.1137/S003614450342480. doi:10.1137/S003614450342480.

Newman, M.E., Girvan, M.. Finding and evaluating community structure in networks. Physical review E 2004;69(2):026113. URL: https://doi.org/10.1103/PhysRevE.69.026113. doi:10.1103/PhysRevE.69.026113.

Ng, A., et al. Sparse autoencoder. CS294A Lecture notes 2011;72(2011):1–19.

Nogueira, S.. Quantifying the stability of feature selection. The University of Manchester (United Kingdom), 2018. URL: http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.740328.

Opitz, D.W.. Feature selection for ensembles. AAAI/IAAI 1999;379:384. URL: http://www.aaai.org/Library/AAAI/1999/aaai99-055.php.

Poggi, J.M., Tuleau, C.. Classification supervisée en grande dimension. application à l'agrément de conduite automobile. Revue de Statistique Appliquée 2006;54(4):41–60.

Preece, S.J., Goulermas, J.Y., Kenney, L.P., Howard, D.. A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data. IEEE Transactions on Biomedical Engineering 2008;56(3):871–879. URL: https://doi.org/10.1109/TBME.2008.2006190. doi:10.1109/TBME.2008.2006190.

Provost, F.F.. The case against accuracy estimation for comparing classifiers. In: Proceedings of the Fifteenth International Conference on Machine Learning. 1998. .

Pudil, P., Novovičová, J., Choakjarernwanit, N., Kittler, J.. Feature selection based on the approximation of class densities by finite mixtures of special type. Pattern Recognition 1995;28(9):1389–1398. URL: https://doi.org/10.1016/0031-3203(94)00009-B. doi:10.1016/0031-3203(94)00009-B.

Quinlan, J.R.. Induction of decision trees. Machine learning 1986;1(1):81–106. URL: https://doi.org/10.1023/A:1022643204877. doi:10.1023/A:1022643204877.

Rakotomamonjy, A.. Variable selection using svm-based criteria. Journal of machine learning research 2003;3(Mar):1357–1370. URL: http://jmlr.org/papers/v3/rakotomamonjy03a.html.

Robnik-Šikonja, M., Kononenko, I.. Theoretical and empirical analysis of relieff and rrelieff. Machine learning 2003;53(1):23–69.

Roelofs, R., Shankar, V., Recht, B., Fridovich-Keil, S., Hardt, M., Miller, J., Schmidt, L.. A meta-analysis of overfitting in machine learning. Advances in Neural Information Processing Systems 2019;32:9179–9189. URL: https://proceedings.neurips.cc/paper/2019/hash/ee39e503b6bedf0c98c388b7e8589aca-Abstract.html.

Roweis, S.T., Saul, L.K.. Nonlinear dimensionality reduction by locally linear embedding. science 2000;290(5500):2323–2326. URL: http://science.sciencemag.org/content/290/5500/2323. doi:10.1126/science.290.5500.2323.

Rudin, C.. Please stop explaining black box models for high stakes decisions. arXiv preprint arXiv:181110154 2018;1.

Saeys, Y., Abeel, T., Van de Peer, Y.. Robust feature selection using ensemble feature selection techniques. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer; 2008. p. 313–325. URL: https://doi.org/10.1007/978-3-540-87481-2_21. doi:10.1007/978-3-540-87481-2_21.

Saeys, Y., Inza, I., Larranaga, P.. A review of feature selection techniques in bioinformatics. bioinformatics 2007;23(19):2507–2517. URL: https://doi.org/10.1093/bioinformatics/btm344. doi:10.1093/bioinformatics/btm344.

Safavian, S.R., Landgrebe, D.. A survey of decision tree classifier methodology. IEEE transactions on systems, man, and cybernetics 1991;21(3):660–674. URL: https://doi.org/10.1109/21.97458. doi:10.1109/21.97458.

Salzberg, S.L.. On comparing classifiers: Pitfalls to avoid and a recommended approach. Data mining and knowledge discovery 1997;1(3):317–328. URL: https://doi.org/10.1109/34.574797. doi:10.1109/34.574797.

Sandri, M., Zuccolotto, P.. Variable selection using random forests. In: Data analysis, classification and the forward search. Springer; 2006. p. 263–270. URL: https://doi.org/10.1007/3-540-35978-8_30. doi:10.1007/3-540-35978-8_30.

Sebban, M., Nock, R.. A hybrid filter/wrapper approach of feature selection using information theory. Pattern recognition 2002;35(4):835–846. URL: https://doi.org/10.1016/S0031-3203(01)00084-X. doi:10.1016/S0031-3203(01)00084-X.

Senawi, A., Wei, H.L., Billings, S.A.. A new maximum relevance-minimum multicollinearity (mrmmc) method for feature selection and ranking. Pattern Recognition 2017;67:47–61.

Sheikhpour, R., Sarram, M.A., Gharaghani, S., Chahooki, M.A.Z.. A survey on semi-supervised feature selection methods. Pattern Recognition 2017;64:141–158. URL: https://doi.org/10.1016/j.patcog.2016.11.003. doi:10.1016/j.patcog.2016.11.003.

Siedlecki, W., Sklansky, J.. A note on genetic algorithms for large-scale feature selection. In: Handbook of pattern recognition and computer vision. World Scientific; 1993. p. 88–107. URL: https://doi.org/10.1142/9789814343138_0005. doi:10.1142/9789814343138_0005.

Sima, C., Attoor, S., Brag-Neto, U., Lowey, J., Suh, E., Dougherty, E.R.. Impact of error estimation on feature selection. Pattern Recognition 2005;38(12):2472–2482. URL: https://doi.org/10.1016/j.patcog.2005.03.026. doi:10.1016/j.patcog.2005.03.026.

Skurichina, M., Duin, R.P.. Combining feature subsets in feature selection. In: International Workshop on Multiple Classifier Systems. Springer; 2005. p. 165–175. URL: https://doi.org/10.1007/11494683_17. doi:10.1007/11494683_17.

Somol, P., Novovičová, J.. Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality. IEEE Transactions on Pattern Analysis and Machine Intelligence 2010;32(11):1921–1939.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research 2014;15(1):1929–1958. URL: http://dl.acm.org/citation.cfm?id=2670313.

Stańczyk, U., Jain, L.C.. Feature selection for data and pattern recognition. Springer, 2015. URL: https://doi.org/10.1007/978-3-662-45620-0. doi:10.1007/978-3-662-45620-0.

Strobl, C., Boulesteix, A.L., Zeileis, A., Hothorn, T.. Bias in random forest variable importance measures: Illustrations, sources and a solution. BMC bioinformatics 2007;8(1):1–21. URL: https://doi.org/10.1186/1471-2105-8-25. doi:10.1186/1471-2105-8-25.

Sun, Q., Pfahringer, B.. Bagging ensemble selection. In: Australasian Joint Conference on Artificial Intelligence. Springer; 2011. p. 251–260. URL: https://doi.org/10.1007/978-3-642-25832-9_26. doi:10.1007/978-3-642-25832-9_26.

Sun, Y., Li, J.. Iterative relief for feature weighting. In: Proceedings of the 23rd international conference on Machine learning. 2006. p. 913–920. URL: https://doi.org/10.1145/1143844.1143959. doi:10.1145/1143844.1143959.

Sutton, R., Barto, A.. Reinforcement learning: An introduction. Computer Science Department, Reinforcement Learning Lectures 2018;.

Tang, J., Alelyani, S., Liu, H.. Feature selection for classification: A review. Data classification: Algorithms and applications 2014;:37URL: http://www.crcnetbase.com/doi/abs/10.1201/b17320-3.

Tenenbaum, J.B., De Silva, V., Langford, J.C.. A global geometric framework for nonlinear dimensionality reduction. science 2000;290(5500):2319–2323. doi:DOI:10.1126/SCIENCE.290.5500.2319.

Tibshirani, R.. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological) 1996;58(1):267–288. URL: https://doi.org/10.1111/J.2517-6161.1996.TB02080.X. doi:10.1111/J.2517-6161.1996.TB02080.X.

Trier, Ø.D., Jain, A.K., Taxt, T.. Feature extraction methods for character recognition-a survey. Pattern recognition 1996;29(4):641–662. URL: https://doi.org/10.1016/0031-3203(95)00118-2. doi:10.1016/0031-3203(95)00118-2.

Van Rossum, G., Drake Jr, F.L.. Python tutorial. volume 620. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

Viswanath, P., Kumar, P.V., Babu, V.S., Kumar, M.V.. Generalized branch and bound algorithm for feature subset selection. In: International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007). IEEE; volume 2; 2007. p. 214–218. URL: https://doi.org/10.1109/ICCIMA.2007.210. doi:10.1109/ICCIMA.2007.210.

Wang, H., Hong, M.. Supervised hebb rule based feature selection for text classification. Information Processing & Management 2019;56(1):167–191. URL: https://doi.org/10.1016/j.ipm.2018.09.004. doi:10.1016/j.ipm.2018.09.004.

Wang, H., Liu, S.. An effective feature selection approach using the hybrid filter wrapper. International Journal of Hybrid Information Technology 2016;9(1):119–128. URL: https://doi.org/10.14257/IJHIT.2016.9.1.11. doi:10.14257/IJHIT.2016.9.1.11.

Weston, J., Elisseeff, A., Schölkopf, B., Tipping, M.. Use of the zero norm with linear models and kernel methods. The Journal of Machine Learning Research 2003;3:1439–1461. URL: http://jmlr.org/papers/v3/weston03a.html. doi:10.5555/944919.944982.

Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., Vapnik, V.. Feature selection for svms 2000;URL: https://proceedings.neurips.cc/paper/2000/hash8c3039bd5842dca3d944faab91447818-Abstract.html.

Wilson, D.R., Martinez, T.R.. Reduction techniques for instance-based learning algorithms. Machine learning 2000;38(3):257–286. URL: https://doi.org/10.1023/A:1007626913721. doi:10.1023/A:1007626913721.

Woeginger, G.J.. Exact algorithms for np-hard problems: A survey. In: Combinatorial optimization—eureka, you shrink! Springer; 2003. p. 185–207. URL: https://doi.org/10.1007/3-540-36478-1_17. doi:10.1007/3-540-36478-1_17.

Xu, Y., Goodacre, R.. On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. Journal of Analysis and Testing 2018;2(3):249–262. URL: https://doi.org/10.1007/s41664-018-0068-2. doi:10.1007/s41664-018-0068-2.

Xu, Z., King, I., Lyu, M.R.T., Jin, R.. Discriminative semi-supervised feature selection via manifold regularization. IEEE Transactions on Neural networks 2010;21(7):1033–1047. URL: https://dblp.org/rec/journals/tnn/XuKLJ10. doi:10.1109/TNN.2010.2047114.

Yadav, S., Shukla, S.. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In: 2016 IEEE 6th International conference on advanced computing (IACC). IEEE; 2016. p. 78–83. doi:10.1109/IACC.2016.25.

Yang, F., Mao, K.. Robust feature selection for microarray data based on multicriterion fusion. IEEE/ACM Transactions on Computational Biology and Bioinformatics 2010;8(4):1080–1092.

Yang, Y., Pedersen, J.O.. A comparative study on feature selection in text categorization. In: Icml. Nashville, TN, USA; volume 97; 1997. p. 35.

Ying, X.. An overview of overfitting and its solutions. In: Journal of Physics: Conference Series. IOP Publishing; volume 1168; 2019. p. 022022. URL: https://doi.org/10.1088/1742-6596%2F1168%2F2%2F022022. doi:10.1088/1742-6596/1168/2/022022.

Yu, L., Liu, H.. Efficient feature selection via analysis of relevance and redundancy. J Mach Learn Res 2004a;5:1205–1224. URL: http://jmlr.org/papers/volume5/yu04a/yu04a.pdf.

Yu, L., Liu, H.. Efficient feature selection via analysis of relevance and redundancy. J Mach Learn Res 2004b;5:1205–1224. URL: http://jmlr.org/papers/volume5/yu04a/yu04a.pdf.

Yu, L., Liu, H.. Redundancy based feature selection for microarray data. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. 2004c. p. 737–742. URL: https://doi.org/10.1145/1014052.1014149. doi:10.1145/1014052.1014149.

Zhang, Z.. Introduction to machine learning: k-nearest neighbors. Annals of translational medicine 2016;4(11). URL: https://doi.org/10.21037/atm.2016.03.37. doi:10.21037/atm.2016.03.37.

Zhao, Z., Liu, H.. Semi-supervised feature selection via spectral analysis. In: Proceedings of the 2007 SIAM international conference on data mining. SIAM; 2007. p. 641–646. URL: https://doi.org/10.1137/1.9781611972771.75. doi:10.1137/1.9781611972771.75.

Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., Liu, H.. Advancing feature selection research. ASU feature selection repository 2010;:1–28.

Zhao, Z.A., Liu, H.. Spectral feature selection for data mining. Taylor & Francis, 2012.

Zhu, X., Goldberg, A.B.. Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning 2009;3(1):1–130. URL: https://doi.org/10.2200/S00196ED1V01Y200906AIM006. doi:10.2200/S00196ED1V01Y200906AIM006.