Adaptive system for analysis process design assistance

Gabriel Ferrettini

▶ **To cite this version:**

Gabriel Ferrettini. Adaptive system for analysis process design assistance. Computer Science [cs]. Université Toulouse Capitole, 2021. English. NNT : . tel-03412695

HAL Id: tel-03412695

https://ut3-toulouseinp.hal.science/tel-03412695v1

Submitted on 3 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur : ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite de ce travail expose à des poursuites pénales.

Contact : portail-publi@ut-capitole.fr

LIENS

Code la Propriété Intellectuelle – Articles L. 122-4 et L. 335-1 à L. 335-10
Loi n° 92-597 du 1er juillet 1992, publiée au *Journal Officiel* du 2 juillet 1992

http://www.cfcopies.com/V2/leg/leg-droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse Capitole

École doctorale : **Droit et Science Politique**

Présentée et soutenue par

Ferrettini Gabriel

le 30/03/2021

Système adaptatif pour l'aide à la conception de processus d'analyse

Discipline : Informatique

Spécialité : Informatique et télécommunication

Unité de recherche : **IRIT: Institut de Recherche en Informatique de Toulouse**

Directeur de thèse : Mme Chantal Soulé-Dupuy, PREX, Université Toulouse 1 Capitole

JURY

Rapporteurs	M. Patrick Gallinari, Professeur des universités, LIP6 Sorbonne Université M. Mohand-Saïd Hacid, Professeur des universités, Université Claude Bernard Lyon1
Suffragants	M. Christophe Claramunt, Professeur des universités, BCRM Brest Ecole Navale M. Max Chevalier, Professeur des universités, Université Toulouse 3 Paul Sabatier
Directeur(s) de thèse	Mme Chantal Soulé-Dupuy, PREX, Université Toulouse 1 Capitole M. Julien Aligon, Maître de conférences, Université Toulouse 1 Capitole

Résumé

Ces dernières années ont vu une croissance rapide de la quantité de données stockées par l'humanité. De nombreux domaines d'activité ont bénéficié de cette croissance en exploitant et en analysant ces nouvelles masses d'informations. Ces nouvelles activités se sont traduites par une augmentation du besoin en capacité d'analyse et de traitement. Malheureusement, l'analyse de données reste une activité ardue et opaque, qui nécessite souvent l'intervention d'outils spécialisés et d'experts formés dans leur utilisation. Or, un utilisateur potentiel de l'analyse de données n'a pas forcément les moyens de s'offrir de tels services, ni le temps de se former complètement à l'analyse. Il apparaît donc que des outils d'analyse de données plus accessibles pour un public plus large seraient bénéfiques à de nombreuses personnes tels que des acteurs de terrain (biologistes, astronomes...).

De cette constatation, découle un verrou principal qu'il nous faut adresser : **Comment pouvons-nous aider un utilisateur à créer son propre modèle d'analyse de données, bien qu'il ne soit pas expert dans ce domaine ?** En analysant les manques de l'état de l'art et ce qui a déjà été fait pour adresser ce verrou, nous le divisons en deux sous-verrous plus précis et davantage centrés sur l'apprentissage automatique : **Comment pouvons-nous recommander efficacement une chaîne de traitement d'apprentissage automatique à un tel utilisateur ?** Puis **Comment pouvons-nous aider un tel utilisateur à comprendre et analyser un modèle d'analyse de données ?**.

L'état de l'art nous apporte en partie des solutions à ces verrous. D'abord, en remplaçant le besoin de décision humaine par l'automatisation des processus d'analyse, notamment dans la création de modèles d'apprentissage automatique. Ensuite, par des outils de visualisation présentant les données et leurs caractéristiques de manière à les rendre plus accessibles pour un humain. Le problème principal de ces solutions est qu'elles sont souvent conçues pour assister des experts en analyse de données et ne sont donc pas accessibles à des utilisateurs non-experts. Pourtant, les connaissances de terrain que possède un utilisateur qui a récolté les données utilisées (par exemple un biologiste) peuvent se révéler très utiles lors du processus d'analyse.

Afin de mettre à profit ces connaissances de terrain, nous proposons donc des solutions d'aide à l'analyse de données qui mettent à profit ce que l'utilisateur connaît. Tout d'abord, nous décrivons dans ce mémoire de thèse notre système de recommandation de chaîne de traitements d'analyse de données. Celui-ci tient compte des préférences de l'utilisateur quant au type de performances qu'il désire obtenir pour son modèle final et est basé sur les méthodes éprouvées de filtrage collaboratif, qui reposent sur la comparaison du problème de

l'utilisateur à des problèmes passés. Ceci nous permet de garder l'utilisateur impliqué dans le processus de création du modèle. Nous proposons également une méthode permettant de décrire comment un modèle prédictif utilise les attributs des données qui lui sont fournies pour produire une prédiction précise. Cette méthode, appelée méthode d'explication de prédiction additive, attribue un poids à chaque attribut du jeu de données. Ce poids représente l'importance de l'attribut dans la prédiction du modèle. Ainsi, nous utilisons les connaissances qu'a l'utilisateur sur ses données pour lui expliquer comment fonctionne son modèle final. Cette méthode surpasse l'état de l'art en conservant une bonne précision, sans pour autant souffrir d'un temps de calcul trop long. Enfin, nous combinons ces deux méthodes pour créer un cadre plus global d'aide à l'analyse de données. En utilisant nos explications de prédictions, nous permettons à l'utilisateur de juger de l'intérêt des modèles qui lui sont recommandés. En comprenant comment chaque modèle recommandé utilise les données qui lui sont fournies, il peut utiliser ses connaissances de terrain pour les évaluer et prendre des décisions informées quant à la construction de son modèle final. De plus, ces explications sont aussi utilisées pour permettre à l'utilisateur d'explorer plus intuitivement de nouvelles instances de ses données à partir des prédictions de son modèle final.

Si nos propositions apportent des solutions aux verrous de cette thèse, il reste cependant de nombreux aspects à explorer. Par exemple la dimension "bac à sable" du cadre général peut être davantage développée par une personnalisation plus profonde de la chaîne de traitements menant au modèle final : choisir les prétraitements sur les données, paramétrer le modèle en profondeur, mettre en place une boucle d'apprentissage... De même, de nombreuses propositions de l'état de l'art visent à assister à l'analyse de données par la visualisation et l'exploration des données de l'utilisateur. Ces solutions pourraient être améliorées par l'ajout d'explications de prédiction qui, misent en regard avec ces informations font émerger une nouvelle perspective sur ces données. En utilisant l'explication de prédiction de manière intelligente, il devient possible pour un non-expert d'effectuer de nombreuses tâches d'analyse en se reposant sur ses connaissances de terrain plutôt qu'une formation en analyse de données.

Abstract

The last few years have seen rapid growth in the amount of data stored by humanity. Many fields of activity have benefited from this growth by exploiting and analyzing these new masses of information. These new activities have increased the need for analysis and processing capacity. Unfortunately, data analysis remains a difficult and opaque activity, which often requires the intervention of specialized tools and experts trained in their use. However, a potential user of data analysis does not necessarily have the means to afford such services, nor the time to be fully trained in data analysis. Therefore, it appears that data analysis tools that are more accessible to a wider audience would benefit many people.

From this observation emerges the main challenge that we must address: **How can we help a user to create his own data analysis model, although he is not an expert in this field?** By analyzing the lack of the state of the art and what has already been done to overcome this issue, we divide it into two more precise sub-issues, more focused on machine learning: **How can we effectively recommend machine learning workflow to such a user?** and **how can we help such a user understand and analyze a data analysis model?**.

The state of the art partly provides some solutions to tackle these challenges. First, by replacing the need for human decision-making with the automation of analysis processes, in particular in the creation of machine learning models. Then, visualization tools presenting the data and their characteristics to make them more accessible to a human. The main problem with these solutions is that they are often designed to assist data analysis experts and therefore are not accessible to non-expert users. However, the field knowledge of a user who has collected the data used (for example a biologist) can be very useful in the analysis process.

To take advantage of this field knowledge, we offer solutions to help data analysis that takes advantage of what the user knows. First of all, we describe in this document our data analysis workflow recommendation system. This takes into account the user's preferences as to the type of performance he wishes to obtain for his final model and operates on a recommendation architecture based on collaborative filtering, which relies on comparing the user's problem to previous existing problems. Thus, the user is entirely involved in the model creation process. Then, we also propose a method for describing how a predictive model uses the attributes of the dataset to produce a single prediction. This method, known as the additive prediction explanation method, assigns a weight to each attribute of the dataset. This weight represents the importance of the attribute in the prediction of the model. Thus, we use the knowledge that the user has on his data to explain to

him how his final model works. This method surpasses those of the state of the art while maintaining a good precision, without suffering from a too-long computation time. Finally, we combine these two solutions to create a more comprehensive framework for supporting data analysis. By using our prediction explanations, we allow the user to judge the interest of the models that are recommended. By understanding how each model recommended uses the data provided to it, he can use his domain knowledge to assess and make informed decisions about the construction of the final model. Also, these explanations are used to allow the user to more intuitively explore new instances of his dataset while using his final model.

While our proposals provide solutions to the challenges addressed in this thesis, there are still many aspects to explore. For example, the "sandbox" dimension of the general framework can be further developed by allowing a deeper customization of the workflow leading to the final model: Choosing dataset pretreatments, changing model parameters, using a feedback loop... Likewise, many solutions from the state of the art aim to assist a user in analyzing data by providing visualization and exploration methods. Those solutions could be extended by using prediction explanations, which when combined with this enhanced information bring out new insights on the data. By using prediction explanation, it becomes possible for a non-expert to perform many analysis tasks thanks to their field expertise rather than a training in data analysis.

Remerciements

Je remercie tout d'abord mes encadrants qui ont su me guider et me soutenir tout au long de ces trois années. Chantal, pour sa patience et ses encouragements, qui m'ont chaques fois motivé à aller toujours de l'avant, ainsi que ses nombreux conseils. Elle a su faire de moi un vrai chercheur et pour cela je lui dois toute ma gratitude. Julien, pour son dynamisme et sa volonté, il a fait des pieds et des mains pour que cette thèse aille toujours plus loin. Sans lui ce mémoire ne serait pas ce qu'il est.

Je remercie également l'équipe SIG pour son accueil et son enthousiasme. Leurs nombreux avis m'ont permis de considérer des angles d'approche que je n'aurais jamais envisagé sans eux.

Un grand merci à l'équipe de l'INSERM, pour l'intérêt qu'ils ont porté à mes travaux, et tout particulièrement Paul Monsarrat, Philippe Vallet et Cedric Dray. Les longues discussions que j'ai eu avec eux ont été une source d'inspiration constante, et leur assistance a été précieuse pour nos diverses expérimentations.

Merci à William, mon prédécesseur, qui a ouvert la voie de cette thèse, et m'a proposé de prendre sa suite. Sans lui je n'aurais pas pu vivre cette aventure.

Mes remerciements vont aussi au laboratoire KADUCEO, surtout Elodie Escriva et Jean-Baptiste Escoffier, qui ont déjà commencé à appliquer mes travaux et dont les résultats sont éxtrèmement encourageants.

Je remercie ma famille, notamment mes parents Marc et Valérie, qui m'ont soutenu et aidé durant mon cursus.

Enfin, je remercie ma compagne, Anne-Sophie, qui est restée à mes côtés tout au long de cette periode, elle a été mon soutien dans les moments les plus difficile. Son amour et sa présence sont chaques jours une source de joie.

It is the empty space which makes the bowl useful.

Lao Tseu

Dans la vie, rien n'est à craindre, tout est à comprendre.

Marie Curie

Contents

List of Figures

List of Tables

Science and everyday life cannot and should not be separated.

Rosalind Elsie Franklin

Invention, it must be humbly admitted, does not consist in creating out of void but out of chaos.

Mary Wollstonecraft Shelley

Chapter 1

Introduction

1.1 Initial problem

The recent explosion of the quantity of data collected by human society has led to a growing need for data analysis capacity. These analyzes are often requested by persons with important knowledge in their own field of expertise, but who may lack knowledge in data analysis. This situation can lead to the obligation to hire a data science expert or to train oneself in this field, which often entails a significant cost, whether in time or money. This expense is sometimes not even possible, which can lead to imperfect analyzes, as in this anecdote recounted in [RSG16]: A hospital wanting to study the evolution of diseases among its patients, gathered an important volume of data and then used a machine-learning algorithm to analyze these data. They used the results of the analytical model thus created for a while, but it was observed afterward that the algorithm mainly took into account the patient's identification number to determine his state of health. This error of including an identification number in its database and omitting to ignore it during an automatic analysis is a classic error well known by experts in data analysis. However, we can also easily understand that a neophyte could commit it, and not realize the problem for a long time.

On a counterpart, a data analysis expert may not have field knowledge of the data he is analyzing. This can also be a problem for data analysis, as it creates a risk that important parts of the data might be ignored by the data analyst.

Thus, the field knowledge of a data field expert can be really useful to a data analysis process. Despite that, little is done to help data field experts participate in data analysis, most of the proposed solutions preferring to focus solely on data analysis experts.

We are therefore facing a problem encountered by many fields: having a growing need

for data analysis, but not every field expert has the time for learning data analysis techniques. Moreover, not all potential users have the mean of hiring a data analysis expert. Finally, when someone is hired to perform the analysis, the field expert might be cut off during the data analysis process, potentially flawing the result. To address this issue, it is important to propose data analysis methods more accessible to all, and particularly to field experts. That would allow them to analyze their data by themselves or to collaborate with a data analyst and interact with him.

All along this thesis, a data field expert will be described as a "non-expert user", in the sense that he is not an expert in data analysis, his skills relying on his own field of expertise.

1.2 Positioning of this thesis

Given our initial problem, we want to guide a user from the data he wants to analyze to the end product of data analysis: a data analysis model. The main scientific issue that we try to overcome in this thesis can be expressed as this question: **How can we help a user who is not an expert in data analysis at building his own data analysis model?**

To do so, we have to take into account that the user will not be able to learn everything about data analysis, and he will need to be guided through the process of creating a data model. In order to guide someone through this, our proposal has to be able to automatically recommend possible steps to be taken for the user. Hence a sub-issue, which can be considered as a part of our main issue: **How can we efficiently recommend a data analysis workflow to a non-expert user?**

But just being able to recommend workflow steps automatically is not enough to be considered as "guiding the user". Successful guidance would include an important part of explaining what is being done and what is produced to the user. A produced result is useful only if the user can have a critical reflection on it. We can see here that our main issue can be divided into another sub-issue: **How can we help a non-expert understand and analyze the results (i.e. the data models) produced by a data analysis process?**

To solve those issues, we will first look into the broad domain of assistance in data analysis, and identify what remains to be done to address our issues.

According to [SO14], the process of data analysis can be divided into five main steps, represented in 1.1:

1. Data collection: gathering data from different sources

2. Data cleaning: Organizing the data in an ordered dataset
3. Exploratory data analysis: The first pass of analyzing the data with simple methods, the results of this analysis could trigger another pass of data cleaning
4. Modeling and algorithms: Second analysis, producing a data model
5. Model exploitation: Using the results of the analysis

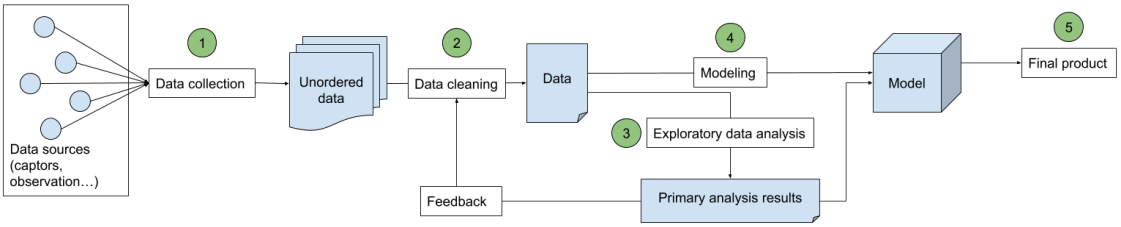


Figure 1.1: The main steps of data analysis

Throughout our exploration of the state of the art, we identify potential lacks mainly in the steps of exploratory data analysis and modeling. Thus, we focused our research on the propositions of the literature pertaining to those two steps.

One of the most common assistance in data modeling is the automated recommendation of workflows. Thus, we look into recommendation systems, and more precisely into those centered on data analysis. Most of the solutions proposed in the literature rely on the comparison of the user's problem to other previous similar problems. This type of recommendation system, called collaborative filtering, is often enhanced by taking into account the user's context and preferences when performing the recommendations. When looking into data analysis recommendation systems proposed in the literature, we notice that this enhancement is very rarely done in the context of data analysis. Thus, this is a possibly interesting option that remains to be explored.

Concerning the problem of helping a user understand the data analysis workflow, a lot of solutions in the form of visualization methods exist in the literature. Most of them focus on representing the steps of data cleaning and preliminary exploration. The domain of the visualization of data analysis models is more recent, and as such, the proposed solutions are not always as satisfactory. The main goal of those proposals aims to counter the black box aspect of data analysis models. For this, diverse authors propose an important number of solutions for explaining a model's inner working globally. A less explored area is the one of explaining single predictions of data analysis models. Whereas single prediction explanations have a real interest for many field expert users who might be more interested in the results obtained for a particular instance, rather than seeing a global analysis.

There still exist many challenges to overcome for explaining the individual predictions of a model: The existing solutions either lack flexibility, precision, or speed. There exist very good solutions but only oriented toward a certain type of model, while the generic solutions tend to pay their genericity either by a low precision or an important computation load.

Finally, concerning the more general field of data analysis assistance framework, we see that most of the proposed solutions are mainly oriented toward data analysis experts. As consequence, a lot of the proposed solutions will tend to be too technical for a non-expert user. Thus, there exists a need for data analysis assistance more friendly to non-expert users.

1.3 Contribution

To address the three main issues previously introduced, we propose multiple solutions throughout this thesis. Those solutions are gathered together into a novel data analysis assistance framework.

This framework first answers to the issue of **efficiently recommending a data analysis workflow to a non-expert user**. It is done through recommendation by collaborative filtering which consists of analyzing the user's problem and recommending the best solution found for the most similar problem known. This is based on a dissimilarity measure, which indicates the differences between the actual user's data and the data of previous data analysis tasks. Yet, this process greatly excludes the user from the decision-making process, which can lead to a lack of trust in what is being done. That is why we propose a solution for considering the **user's preferences** on which kind of data analysis model he would want to be produced. As an example, a user could prefer a model with as few false positives as possible. Moreover, This prediction recommendation system is used to further improve our framework. An opaque data analysis workflow that has been automatically recommended to a user can become a source of mistrust, as its way of functioning is not known. If a user does not understand how the final model works, there is a risk of misusing it. Hence, we can see a new challenge for **efficiently recommending a data analysis workflow to a non-expert user**. Thus, we want the user to take an active part in the training and building of the model. The recommendation system goal is not to make decisions for him, but rather suggest to him a possible solution from which he can make his choice.

This leads to the issue of helping a user **understand and analyze the results produced by a data analysis process**: as the inner function of a predictive model

is not always accessible, we have to instead use the inputs given to the model and analyze its output. A simple way of representing how a model is using the data of the user to produce its results is by showing the importance each part of the data has for the prediction. This relies on additive explanation systems: when a predictive model produces a prediction from an instance, this prediction is explained through the importance given to each attribute of the instance when making the prediction. This importance is presented to the user as a set of weights, with a higher weight signifying a higher influence of the attribute on the prediction. As we intend to help as many non-expert users as possible, we cannot rely on model-specific methods of explanation. Because of that, we have to rely on model-generic prediction explanation systems. The main problem with model-generic additive prediction explanations is the important computational time needed to achieve a satisfactory precision in the explanation. This important computational weight is caused by the fact that to obtain high precision in the explanation, the method has to take into account the influence of every possible group of attributes of the dataset. This heavy calculation cost leads to a computation time over several hours for a dataset of 10 or more attributes, or even several days for larger datasets. Thus, we propose a method to identify the most relevant groups of attributes that need to be taken into account to achieve a satisfactory level of precision in the explanation. With this method, we can calculate only the most important information, allowing us to strike a balance between precision and computation time when explaining the predictions of a model. Our proposal for prediction explanation is then tested through a large experiment including more than 300 datasets. During this experiment, we use different methods of identifying important information. With this experiment, we then determine the strong and weak points of our different methods depending on the data analyzed and the problem at hand.

Finally, we tackle the larger issue of **helping a user who is not an expert in data analysis at building his own data analysis model**. Looking at the state of the art shows us that a lot of data analysis assistance solutions exist, but very few of them are oriented toward helping non-expert users. In order to address this problem, we decide to use our prediction explanation method in combination with our workflow recommendation. Once the recommendation system has produced a set of possible workflows to apply to the user's data, we explain how each recommended element functions through their prediction explanation. Those explanations give insight to the user, which makes him able to make an informed decision when building his predictive model, rather than blindly following what is being recommended to him. He is then able to choose which workflow he wants to

apply to his data and consider the quality of his data. As the explanations are based on the importance of each attribute, the user can see which attributes are being used during the analysis, and which are being ignored. With this information, the user can decide to remove or add attributes to change the different models' behavior.

Then, the user is guided through the exploitation of his newly trained predictive model. He is asked to create new instances for the model to predict. Those predictions are then explained to him through our explanation system. This aims to give him a sandbox environment where he can explore diverse hypotheses and see their effect on the model's predictions. Through this, the user is given more information from the prediction than the sole model output. This paves the way for discovering new interesting trends and behaviors in new data.

This framework finally needs to be evaluated by creating a prototype from it. The development of a prototype gives us the possibility to get feedback from actual users and create a real-life experiment to test our framework.

1.4 Organisation of this thesis

This document is divided into two main parts. Part I is dedicated to describing our exploration and analysis of the state of the art. Part II describes our contributions toward answering the identified main issues.

Throughout Part I, we first contextualize our work in the greater ensemble of the state of the art literature. In Chapter 2, we take a tour of the different data analysis assistance tools existing, first in the broad domain of general data analysis, and then focusing it on the creation of machine learning models. Then, in Chapter 3, we take a look at the recommendation systems, and more precisely the machine learning workflow recommendation systems. We come to the conclusion that most of the machine learning recommendation systems rely on collaborative filtering, but few are extended through the use of the user's context. The last part of our state of the art (Chapter 4) focuses on the explanation of machine learning models to users. The goal of the model explanation is to overcome the black box aspect of an automatically trained model by illustrating its way of functioning.

After having identified the limits of the proposals of the literature concerning the main issues we identified, we then propose our solutions for addressing these issues in Part II, dedicated to our contributions.

In Chapter 5 we first describe our automated recommendation system which focuses on

taking into account the user's preferences when recommending possible workflows. This recommendation system is then tested to validate its usefulness. The end of this chapter is dedicated to analyzing the results of those tests.

We then tackle the problem of helping the user understand the raw results of an automated recommendation, which leads us to create a solution for explaining model predictions in chapter 6. This proposal is inspired by the additive explanations from the literature but aims to strike a balance between precision and computation speed. Our research process produces many possible explanation methods suitable for diverse situations depending on the dataset of the user and the model built from this dataset. Those systems are then evaluated, and their performances are analyzed.

These two proposals are then combined in a more complete framework in chapter 7. It aims to exploit prediction explanation in a novel way to bring new insight to the user in several steps of the creation of a machine learning model. This framework is then developed into a complete prototype, for its evaluation with different users in real-life situations.

Finally, in chapter 8, we present a summary of our work and what has been achieved during this thesis, but also describe the possible developments and perspectives that we consider for our work.

References

[Abd18] Salisu Mamman Abdulrahman et al. "Speeding up algorithm selection using average ranking and active testing by introducing runtime". In: *Machine learning* 107.1 (2018), pp. 79–108.

[LBV12] Rui Leite, Pavel Brazdil, and Joaquin Vanschoren. "Selecting classification algorithms with active testing". In: *International workshop on machine learning and data mining in pattern recognition*. Springer. 2012, pp. 117–131.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. event-place: San Francisco, California, USA. New York, NY, USA: ACM, 2016, pp. 1135–1144.

[SO14] Rachel Schutt and Cathy O'Neil. *Doing data science: Straight talk from the frontline*. O'Reilly, 2014.

[Van13] Joaquin Vanschoren et al. “OpenML: Networked Science in Machine Learning”. In: *SIGKDD Explorations* 15.2 (2013). Place: New York, NY, USA Publisher: ACM, pp. 49–60. DOI: 10.1145/2641190.2641198.

Part I

Exploring assistance in data analysis : State of the art

Preamble

In recent times, machine learning has received a growing interest from domains always more numerous. This growth is well illustrated by the increase in researches of the terms "machine learning" on google, as shown in 1.2. This increased attention is in part centered on machine learning, and its possible applications. Such a surge in attention on this domain led to a large number of machine learning tools, but also a plethora of machine learning assistance tools, aiming to facilitate the creation and training of a model from any data. Despite this, machine learning is still a largely opaque domain, with most of the work being done by data analysis experts. This is a deterrent to the use of machine learning tools, as it can be expensive to hire a data analysis expert, and not everyone has the time required to become one. This means that there is still a need for further assistance in data analysis. To understand what needs to be done to help an expert or the general public in performing data analysis and machine learning, we must take a look at what has already been done.

But first, we have to make the distinction between data analysis and machine learning.

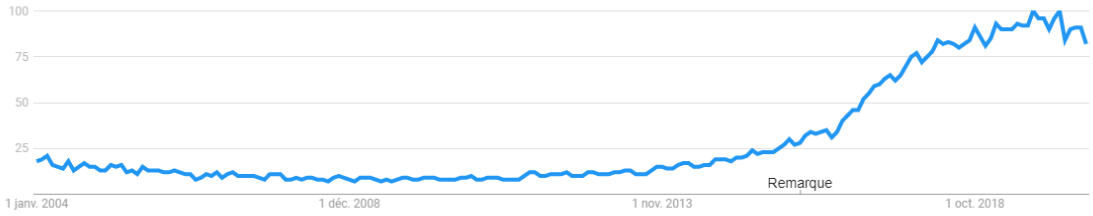


Figure 1.2: Evolution of the number of researches for "machine learning" on google, according to google trend

Here are our definitions of the terms "machine learning" and "data analysis", according to a synthesis of their usage across the state of the art ([Abd18; LBV12; Van13])

Definition 1 *Data analysis is the general domain pertaining to the study of existing data through its cleansing, transforming, and modeling with the goal of discovering useful information, informing conclusions, and supporting decision-making.*

Definition 2 *Machine learning is a particular domain of data analysis. This field is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on a dataset, known as "training data", in order to make predictions or decisions*

without being explicitly programmed to do so.

In short, machine learning is a subfield of data analysis, which aims to create a model from a dataset. This model is created from an algorithm theoretically capable of producing a model from any data, rather than being tailored for a particular task. In this chapter, we first explore the general field of assistance in data analysis, notably through the different tools and techniques that are now available to the general public or in the literature. We establish a survey of what is done in data analysis in Chapter 2. We can conclude that assisting non-expert users is often done by automating parts of the analysis. Then, we decide to focus more on the automatic recommendation of machine learning processes in Chapter 3. This focus allows us to understand a key drawback of assistance in machine learning: the lack of comprehension from the user of the tool he is using. This leads us then to take a look at model explanation, and more precisely, single instance prediction explanation in Chapter 4.

Hey sky, take off your hat, I'm on my way!

Valentina Tereshkova

The purpose of a storyteller is not to tell you how to think, but to give you questions to think upon.

Brandon Sanderson

Chapter 2

Assistance in data analysis

As our goal is to **help a user non-expert in data analysis at building his own data analysis model**, it is primordial to identify what constitutes assistance in data analysis, and what has been developed in this field. In order to do so, we rely on the large number of propositions for assistance in data analysis found across the literature. Moreover, we also study the tools available to the general public the users choose to use, and the way they use them. Those are an important insight into what works and what is lacking. They can function as a compass, pointing to how the users want data analysis assistance to be developed. Thus, we can identify the main themes of data analysis assistance through these tools. In this chapter, we first identify the main forms of assistance in data analysis in Section 2.1. From this identification, we recognize a lack of assistance in the field of machine learning, and so we decide to focus on assistance in this field in Section 2.2. From this domain, we identify two main forms of assistance in machine learning. The first one is the assistance in understanding the machine learning process and described in 2.3. The second one is the assistance in building a data analysis model through machine learning which we discuss in 2.4.

2.1 The main forms of assistance in data analysis

Data analysis is a process including five main steps according to Schutt et al. in [SO14]. Those five steps can be summarized as such :

1. **Data collection**: Raw data is collected from the world and gathered into a single dataset. The source of this data can be of numerous types, from data automatically collected from an automated sensor to a large repository of messages sent on a social media platform.

2. **Data cleaning**: Once this raw data has been collected, it must be processed to create a usable dataset. This process aims most of the time to create a two-dimensional array, with each line being an instance of the data, and each column a characteristic of those instances.

3. **Exploratory data analysis**: This is the first instance of analyzing the data in the data analysis process. This process is done to highlight the main characteristics of the data collected, and give the analyst a clearer sense of what the data contains. This process can also highlight eventual lacks in step (2), prompting further cleaning of the data.

4. **Modeling and algorithms**: The data is now processed in order to answer the original problem that prompted the need for data analysis. It could be, as stated in [SO14], a classification problem, a prediction problem, or a simple description problem, for instance. At the end of this step, a model responding to the main problem is obtained.

5. **Data product or communication**: This step consists of actually using the results of the data analysis. This analysis might have produced a "data product", such as a recommendation system, a weather forecast model, or any other algorithm based on the results of the analysis. It could also be a simple report to the persons who have asked for the data analysis in the first place, giving them an understandable form of the results obtained.

Digital tools are available in the literature to help for each of those steps, with the exception of data product or communication which is purely a synthesis work or porting the results of the analysis to a computer application.

2.1.1 Data collection

Data collection has been made easier through the science of data mining, which has been automated by a large number of tools as clustering data mining, a lot of which are described in [Ber06]. The goal of clustering data mining is to gather a large pool of data and regroup it in different categories through automated patterns and similarities recognition. The survey [GG99] also lists a lot of data mining and knowledge discovery techniques that have since been tested and are largely used in the general public, such as Weka ([Hal09]). In general, those tools facilitate the gathering of data from the large data pool that is the internet. The rest of the data gathering is often facilitated by automatic detection and

classification of subjects of interest for captors. As an example, [Spa08] describes a system for the automatic counting and tracking of fishes in underwater videos. This kind of system assists the user in gathering new data without having to do it by hand. In general, data gathering today has largely been researched and is often only constrained by the size of the data pool from which to gather data, which led to the emergence of the domain of big data analysis.

2.1.2 Data cleaning

As data cleaning is often done with one specific data analysis task in mind, a lot of the process is automated through tools aimed at this specific task ([Kri16]). Yet, some steps are sufficiently generalized to be considered as *general* data cleaning tasks. Among those are the elimination of absurd outliers and the selection of pertinent attributes for the task at hand. A lot of data analysis tools propose to aid a potential user in performing those, such as Orange, What-if, or Weka ([Dem13; Wex19; Hal09]). Moreover, many database management systems have been developed to organize, manage, and manipulate collected data. A recent survey described multiple categories of data management systems in NoSQL [Ges17]. Those systems allow for an easier and faster analysis of the data, depending on the situation of the user.

2.1.3 Exploratory data analysis

It is during this step that a lot of well developed basic methods are applied in order to perform a first exploration of the data. Those methods include, as an example, a linear regression of the data or a principal component analysis. Those steps have been automated through many methods implemented in a large number of tools available to the public, as Matlab ([MAT10]). An example of exploratory analysis is the OLAP (Online Analytical Processing) systems as Xplenty ([NMY14]) or IBM Cognos ([Adk13]). Those systems aim to help users explore multidimensional data through a detailed visual interface and diverse analysis tools. They often rely on recommendation systems in order to automatize a part of the exploration, as in [Gia09]. According to [SO14], this process may reveal a lack of the previous data cleaning process, prompting a new data cleaning before progressing further.

2.1.4 Modeling and algorithms

This last step is the one that produces what will become the final product of the analysis. This is where the field of machine learning is very important as it developed in order

to create models from data automatically via a generic algorithm. Those algorithms are described in greater detail in Section 2.2. Here, machine learning is still largely studied, with, as an example, the advent of neural networks which opened new possibilities in machine learning.

2.1.5 An open field

The different steps of data analysis have been largely explored and automatized. A data scientist has a great array of tools and algorithms at his disposal when performing data analysis. Yet, there seems to be a lot to discover in many fields, particularly on modeling, for which new possibilities are discovered frequently. Yet, as it is a relatively new field when compared to the other aids in data analysis, the tools aiming to assist a user in machine learning are less developed and more experimental. Thus, there is potential in a further study of assistance in machine learning.

2.2 The main forms of assistance in machine learning

When building this state of the art, we have identified three main ways of helping a user create and train a machine learning model:

- First, tools like the ones proposed by [Hal09], which use visual tools to assist data analysis experts in performing data analysis and machine learning tasks. These tools can be called "*Comprehension assistance*" tools, as they do not aim to automatize any process, but rather try to clarify the information by displaying it in a new way or show novel pertinent information that can be inferred from the data.

- The second main category of tools is the "*process automation*" tools, which aim to rely as few as possible on human decisions, by automating important parts of the process of building a model from a dataset. One example can be found in [Feu19], which presents a framework for automatically building a machine learning model, from the selection of the workflow to the final training of the model. Those tools are described in more detail in Chapter 3.

- The third category can be found in [HK13], as an example. Rather than fully automatizing a data analysis process, these tools guide the user through the construction and training of a model, without replacing him entirely. This is often done via visual interfaces depicting the possible actions of the user and the results of these actions.

It can be described as a combination of *comprehension assistance* and *process automation*, from which arise a new possible application that we name *assistance in model building* in this thesis.

Through these three categories, we review the diverse applications and tools developed in the literature. First, we take a peek at two primaries examples:

One of the best-known data analysis assistance tool is weka ([Hal09]), created in 1992, and regularly updated ever since. In its current version, weka offers a large array of functionalities that can be roughly separated into two main categories: data-based and model-based functionalities. In terms of data, weka provides visualization tools to allow the user to better explore their datasets, through their experimentation tool. Moreover, this tool allows the clustering and prepossessing of the data, before the application of basic classification models. More advanced machine learning tools are proposed in the knowledgeflow section of weka, where the user is recommended data analysis workflows and other processes automatically.

More recently, Google released in 2019 its tool "What-if?" ([Wex19]). This tool is designed as an expansion of jupyter, Colaboratory, and Cloud AI Platform notebooks. What-if mainly focuses on helping a user understands his data and, more importantly, the models produced by the data analysis. Some of the proposals are similar to Weka, but the main originality of the What-if tool rests in the possibility of testing hypotheses, by manipulating data on the fly and seeing the effects of the manipulation on both the dataset and the prediction of the trained models. Moreover, What-if also proposes automation of workflow choice and model construction, but those are more akin to the rest of the state of the art.

In order to understand what can be done to further assist in data analysis, we need to review what has already been done. Through this review of the state of the art in data analysis, we need to consider each category of machine learning assistance tools in turn.

2.3 Comprehension assistance

Because of this need for transparency, the existing systems and toolkits for machine learning (ML) and data analysis in general mostly focus on providing and explaining the *methods* and *algorithms*. This approach has proven particularly helpful for expert users, but still requires advanced knowledge of data analysis. Indeed, some of the most well-known data analysis platforms such as Weka [Hal09] or Knime [Ber07] provide detailed descriptions

of the methods and algorithms they include, often giving usage examples. Unfortunately, detailed descriptions are a poor substitute for actual training in data analysis.

Some data analysis platforms, such as RapidMiner ([HK13]) or Orange ([Dem13]), for instance, have dedicated great attention to the problem of presenting and explaining the analysis *results* to the user. By providing well-designed visualization interfaces, these platforms assist the user in understanding the results produced, which is a first step toward actually using them and acting on them. As an example, Orange allows us to study in great detail the results produced by different machine learning models.

Example 1 *In figure 2.1, the user has trained three models on a dataset: a Support Vector Machine, a Naive Bayes, and a Random Forest. These models produced errors when classifying instances of the dataset. These errors are displayed on the right of the screen. They are then collected together and displayed in the form of a Venn diagram, which informs the user of the most commonly misclassified instances of the dataset, depending on the model.*

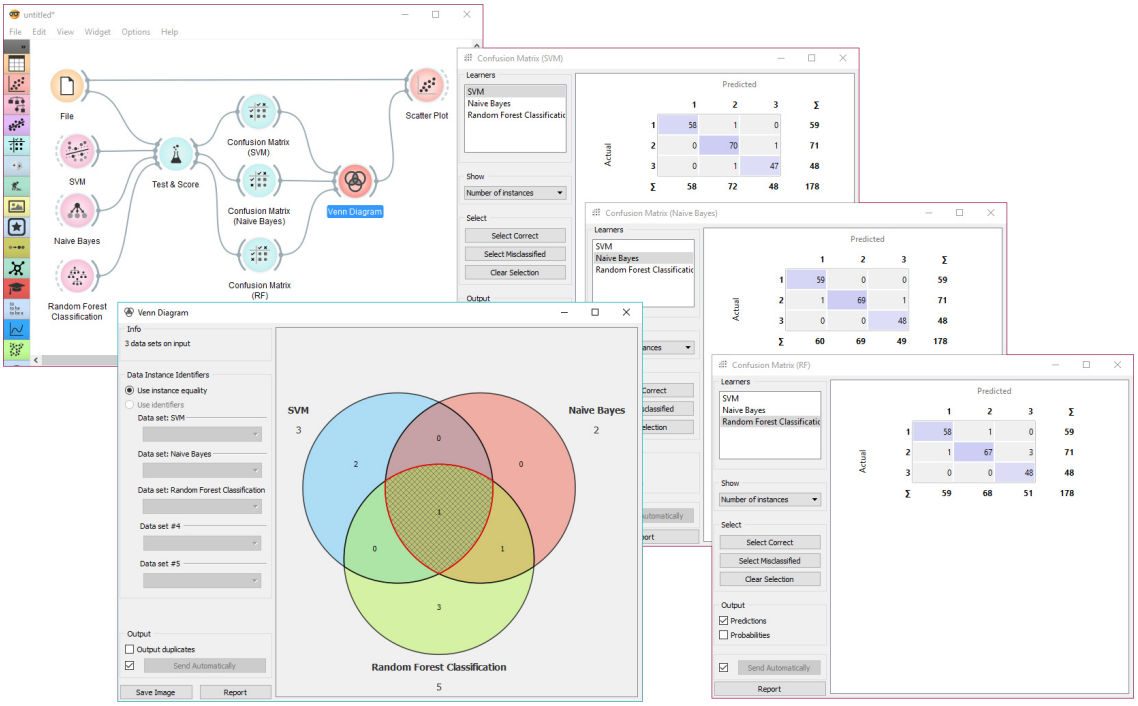


Figure 2.1: Orange : Displaying the venn diagram of the instances of a dataset misclassified by three different models

However, these techniques cannot explain *how* these specific results have been achieved, which remains a significant disincentive for users in areas where wrong decisions can have grave consequences. Helping those users to grasp *why* a particular prediction is being made (in a way that would allow them to check this reasoning against their expert knowledge) could greatly enhance their *trust* in a reasonable prediction, or on the contrary, give them

a meaningful reason to discard a biased one. This is the original intuition behind the need for prediction explanations.

More recently Google with its "What-if" tool[1], mainly based on [WMR17], proposes many exploratory machine learning tools. Those help a user to understand and exploit machine learning models in an intuitive way. This is mainly done by allowing the user to explore new data points with a trained model, and displaying different metrics in an easily interpretable way.

Example 2 *Figure 2.2 depict one of those tools: the user can apply hypotheses on his dataset, either on a single data point or on the whole dataset. As an example, the mean of a particular feature could be modified. Those changes are then reported and displayed to the user. The results of the modifications are shown to the user, both in the dataset and on the classifications of the machine learning model.*

Another of these What-if tools is the model performance visualization tool, depicted in Figure 2.3. This tool displays diverse performance metrics to the user for his machine learning model. The user can then modify the parameters of the model and directly see the impact of those modifications on the performance of the model.

Other visual functionalities of the What-if tool include the capacity to test hypotheses on a dataset and its corresponding trained model. The user has access to a lot of options for modifying artificially the dataset, either through single modifications (replacing an instance by another) or through general changes (changing the mean of an attribute or changing its standard deviation as an example). Those changes are then displayed to the user through visual interfaces similar to the one displayed in Figures 2.2 and 2.3. This way, the user can observe the results of his hypotheses both on the dataset and the classification results of the model. Those options answer to the problems raised by [Haa11]. In this paper, the authors formulate a need for applying hypotheses to a model rather than just using it as a description of a situation.

2.4 Assistance in model building

2.4.1 Visual model construction

Along with visual aids, other tools chose to guide the user in the different steps of workflow building, without suppressing his participation. Rapidminer and other tools such as Magic

[1] https://pair-code.github.io/what-if-tool/

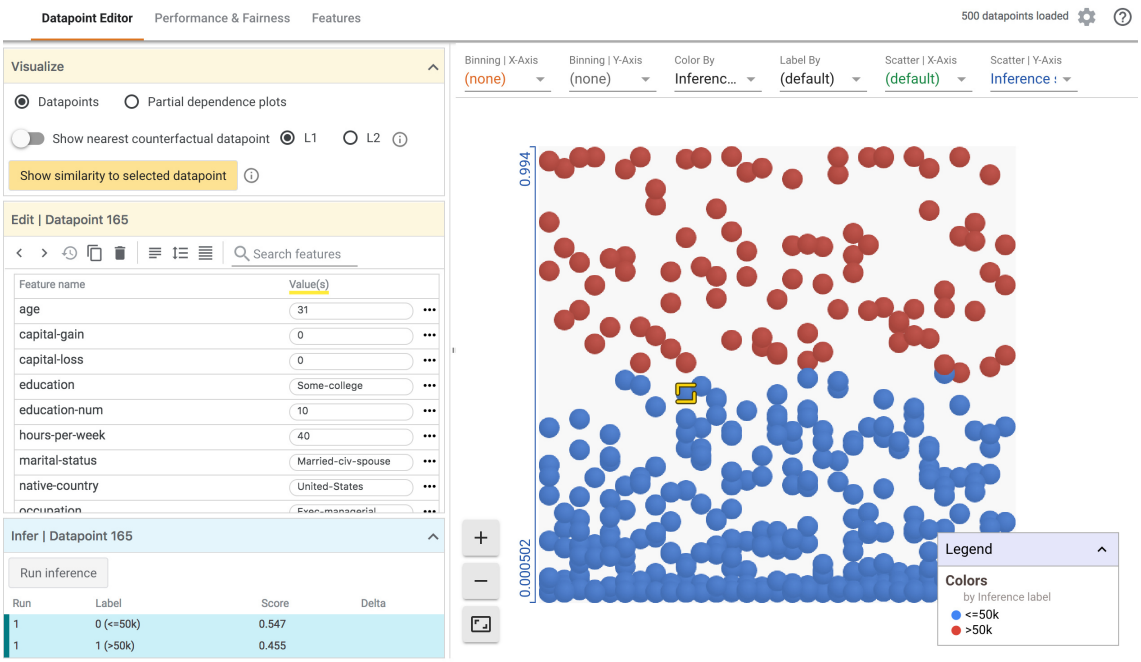


Figure 2.2: Modifying a data point and seeing the results on the model and the dataset in the What-if tool

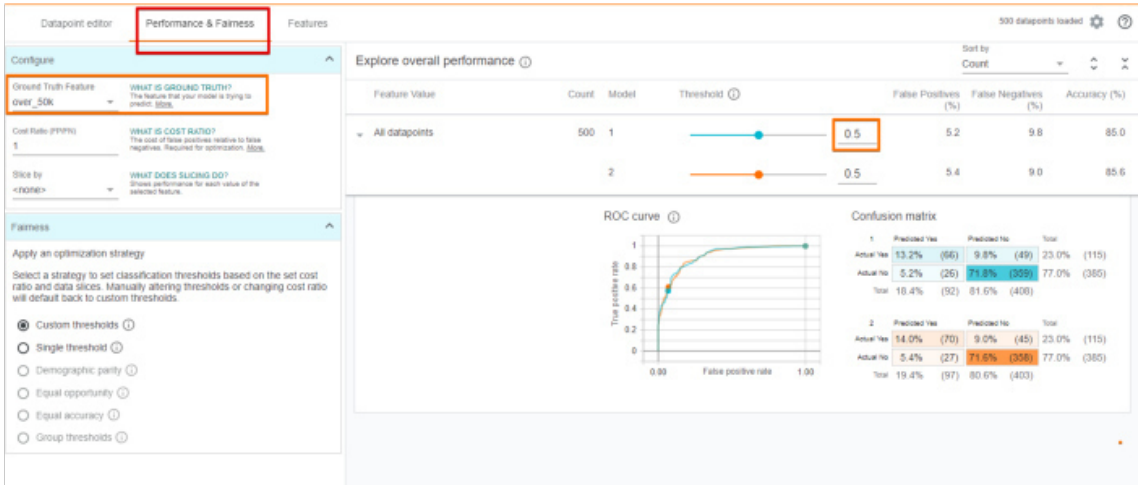


Figure 2.3: What-if tool model performance presentation

quadrant ([HK13; Ido18]) proposing visual model building solutions tend to try to achieve this goal.

Example 3 *In figure 2.4, we can see the rapidminer's visual workflow building tool. Depicted on the picture are 4 steps of a workflow: First, the data is retrieved, it is then assigned roles depending on the nature of the data. The next step consists of converting part of the data from numerical to binomial. The last step is the cross-validation of the future model.*

The main problem with the "building blocks" proposed by the integrated recommenders is that they tend to stay obscure to the non-expert user.

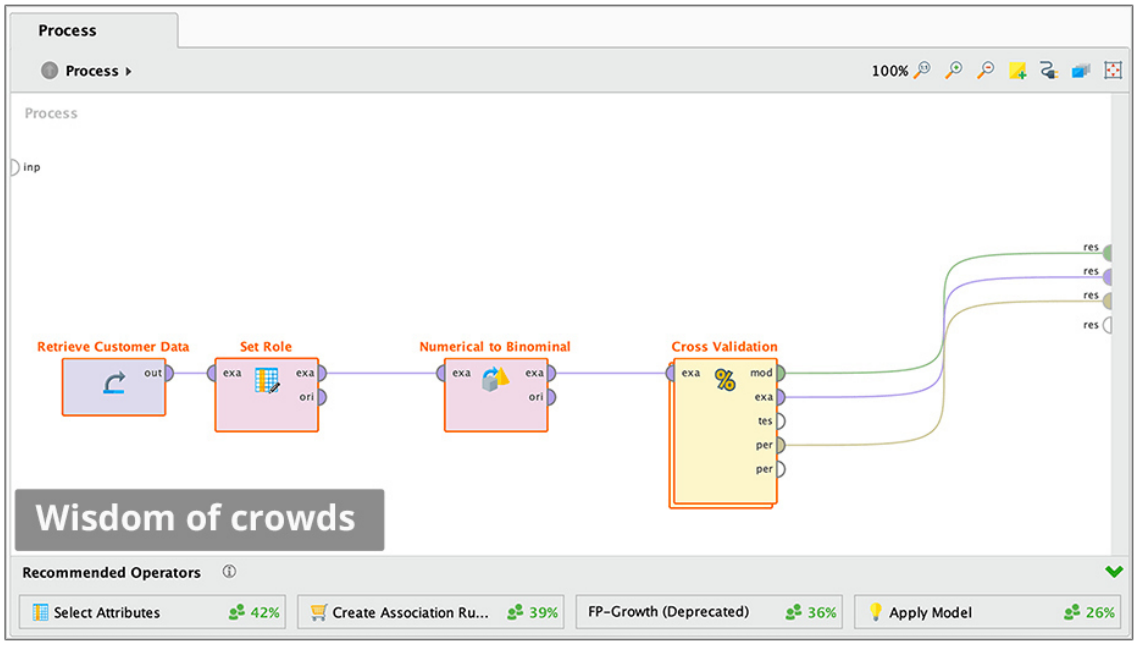


Figure 2.4: Rapidminer's visual workflow designer

2.4.2 Automated optimization

Another form of assistance in model building is the automation of the hyperparameters of a chosen model. Most of the machine learning models do not come in the form of a single monolithic bloc, but rather as a tool with a lot of possible customization and many optimizations. As an example, a neural network can exist in many possible forms, depending on the number of nodes constituting it and its organization. Those modifiable characteristics of a model are called hyperparameters.

To simplify the usage of those models, some authors propose a solution for automatically choosing the model hyperparameters. Those systems often rely on previous usages of the model, in order to determine its best configuration given the task at hand.

As an example, [Feu19] proposes to automatically tune the hyperparameters of a model by creating a "meta-model": Given a database of past machine learning experiments, their system relies on learning from the performances of each models depending on their hyperparameters and the dataset being analyzed. With this base of knowledge, they create a predictive model capable of predicting the hyperparameters most likely to produce good results when given a new data analysis task. This basic configuration is then tuned further by testing the newly created model on small samples of the dataset and observing the changes of performances provoked by changing different hyperparameters. The system then functions as an optimization algorithm, aiming to enhance the performance of the model through the fine-tuning of its hyperparameters, each loop of testing allowing the system to further approach the "ideal" configuration.

2.5 Synthesis

Among these diverse tools integrating data analysis assistance, a lot of them are still aimed at users with a certain expertise in data analysis and machine learning. These tools rely more on a simplification of the expert's work than the guidance of non-expert user through the steps of data analysis.

Table 2.1 summarizes the different approaches that each tool described in this state of the art include in their functionalities. It is evident from here that *visualization tools* are pervasive in the domain, their implantation and utility being evident at this point, as a clearer presentation of information is always preferable. The aid in *feature selection* is also present in virtually every data analysis tool, as the constitution of a well-built dataset is important for a lot of data analysis processes.

Other functionalities appear to be less pervasive, as *workflow recommendation,* (done by Rapidminer and What-if tool) and *workflow building assistance,* (done by Rapidminer but not by What-if tool) are not always provided, and not always in combination. As we have seen, a full workflow recommendation system tends to remove the user entirely from the analysis process, which tends to create distrust from the user. That's why a non-expert user should have access to a combination of workflow recommendations and workflow building assistance. When a user has access to both, the recommendation system can help him take decisions he could not take alone, while the building assistance gives him a more user-friendly interface to interact with.

Finally, very few tools allow their users to *explore the instances* of their datasets through diverse visualization tools. The What-if tool has clearly given a lot of attention to this

approach, which guides the user in exploring their data in new and clearer ways.

The last approach in machine learning assistance we have seen is the explanation of predictions made by a machine learning model. This function has been explored in the literature, as we are going to describe in Chapter 4, but is rarely integrated with a bigger framework. This lack of focus on the comprehension of the models produced by the machine learning processes leads to a need for expertise in machine learning for using these tools correctly. Moreover, the automatization of many processes while building the workflow and the machine learning model leads to a lack of implication of the user in his own model.

In this thesis, we intend to keep the user invested in the building of the machine learning model without requiring detailed expertise in machine learning from him. This can be done by relying on the knowledge the user has of his data, which is often neglected in classical tools.

		Weka	Rapidminer	Knime	Orange	What-if tool
Used approaches	Visualization tools	✓	✓	✓	✓	✓
	Prediction explanation	✓				
	Instances exploration				✓	✓
	Features selection	✓	✓	✓	✓	✓
	Workflow recommendation		✓		✓	✓
	Workflow building assistance		✓	✓	✓	
Relation with the user	User implication	Medium	Medium	Full	Medium	Full
	Data knowledge exploitation	Medium	None	Medium	Medium	Medium
	ML expertise needed	Full	Full	Full	Full	Medium

Table 2.1: Summary of the different data analysis tools and their functionalities, along with their relation to their user: respectively, is the user involved in the process, is his knowledge of the analyzed data exploited, and is a data analysis expertise required.

We ignore public understanding of science at our peril

Eugenie Clark

As always in life, people want a simple answer... And it's always wrong

---

Susan Greenfield

Chapter 3

Recommendation systems

One of the identified issues is the efficient recommendation of a machine learning workflow to a non-expert user. To address this challenge, we first have to take a look at what already exists in the literature, both in the general domain of recommendation systems and in recommendation systems aiming to recommend machine learning processes. As we aim to recommend to persons who are not experts in data analysis, we will have to rely on what they have at their disposal for the recommendation process. Thus, we want to base our recommendation on the data of the user, and his objectives for the analysis. Thus, we first take a look at the collaborative filtering recommendation approaches in Section 3.1, as they rely on comparing the user's problem to a database of previous problems. In Section 3.2, we then focus our attention on recommendation systems precisely aimed at recommending machine learning processes. During this analysis, we realize that a common extension of collaborative filtering systems is not present in machine learning recommendation systems. This extension is the consideration of the user's context when performing the recommendation. Thus, we take a look at context-aware recommendation systems in Section 3.3. Finally, we discuss the problems brought by unexplained recommendations in Section 3.4.

3.1 Collaborative filtering approach

Model recommendation is the basis of a lot of data analysis assistance tools. Most of these tools recommend possible workflows by relying on a database of past experiments, to find solutions used on similar problems. Those are called **collaborative filtering recommendation systems**. The main principle of these recommendation systems is to assess the user's problem (choosing a movie or a restaurant, solving a technical problem,

finding a solution to a computer bug...) and compare his profile to the activities of other users.

Example 4 *Marc is looking for a new movie to watch on his favorite streaming platform and decides to look at the movies recommended to him by the service. Let's say the platform has seven movies. (1): Who framed Roger Rabbit?, (2): Matrix, (3): Django Unchained, (4): What we do in the shadow, (5): Hot fuzz, (6): John wick, (7): Artemis Fowl. The recommendation system of the platform uses the previous movies watched and rated by Marc. These ratings are compared with the ratings of other users, giving a matrix of ratings in Table 3.1.*

movie	*(1)*	*(2)*	*(3)*	*(4)*	*(5)*	*(6)*	*(7)*
Marc		9	10		7		
Lucy	4	8	2				
John	2	8			8	10	
Sophie		10	9			8	3

Table 3.1: ratings of movies by different users of the streaming platform

The system identifies John and Sophie as the two closest users to Marc as they both watched and liked two movies Mark did too. Then, the system sees that John and Sophie both watched and liked John Wick, a film Mark has not seen yet. Thus, Mark is recommended John Wick as a movie he could potentially appreciate.

In our problem, we aim to recommend a workflow to a user based on his dataset and his desired types of results or analysis needs. This is the user profile, as it indicates the user's problem and his preferences. Now, in this case, the comparison between different users to find older users close to the new user is often subject to discussion, and we have to find the solutions proposed in the state of the art.

Recommender systems based on collaborative filtering are known to be effective in various applications. For example, [BHM04] is applied in the context of database research and management. Given a database, the recommender system suggests queries based on previously issued queries. [HC10] describes a system aiming to assist a user by recommending him items that would interest him, based on its interactions with other users in a social network. Another example is the one of [Ali15], which provides sequences of queries based on similarities between *OLAP* user sessions.

3.2 A dive in machine learning recommendation systems

In a lot of cases, it is simpler for a non-expert user to rely on a data analysis expert rather than try to perform his analysis by himself. The main problem of experts is their scarcity and their cost. Thus, a lot of researchers developed many ways to simulate the behavior of an expert by creating automated machine learning processes. The recent survey of Joaquin Vanschoren [Van18] listed many automation techniques under the common name *autoML*, for automated machine learning. Yet a lot of these automated solutions have very different approaches, and often focuses on different points of a model in order to analyze data. The common point of all those techniques is that they are all based on the previous machine learning experiments to infer the performance of different possible models on the task being automatized. Hence, these processes are grouped under the term of *Meta-learning*, meaning that they learn from previous learning tasks. *Meta-learning*, here designate the action of learning from previous data analysis tasks to determine the best course for a new task in order to automatize the totality or a part of the process.

A lot of these tools need an important repository of past machine learning experiments to function. For this, an important data analysis project is the Openml ([Van13], `https://www.openml.org/`) community, which main aim is to gather and store a large number of machine learning experiments and encourage collaborative work between users. This is an original approach as the goal is not to put the user through an automatizing process, but rather to connect him with an expert capable of and interested in helping him. The Openml interface offers a large number of results visualization and comparison: once multiple machine learning processes have been applied to the user's dataset, the results can be viewed in a list organized by different classical evaluation criteria. Moreover, the users can specify the type of data analysis tasks he wants to be performed on the dataset, with parameters allowing him to specify the preferred evaluation criterion and many other specific options. This community also have led to the constitution of a large collection of datasets and machine learning experiments, facilitating a lot of research works as [Feu15] or [DK17].

Moreover, their organization of the process of machine learning allows us to distinguish different elements, which we define here according to their usage on their website.

Definition 3 *A **dataset** is the collection of data on which the analysis will be performed. It can take many forms, but in machine learning, it is often organized as a two-dimensional table. One dimension of the dataset is the collection of the instances that will be studied. As an example, in a medical study, an instance would be an individual patient. The second*

dimension is the ensemble of the attributes of the instances. In our example of a medical study, the attributes would be the characteristics of each patient, such as their weight, age, size, etc. Given $A = a_1, ..., a_n$, the attributes of a dataset, an instance x is a vector of n attributes values: the description of x along the attribute set A.

Definition 4 *A **Workflow** consists of a series of data analysis algorithms and actions leading to the creation of a machine learning model. As an example, a workflow could consist of the elimination of missing values of the studied dataset, then an attribute selection reducing its size, followed by the training of a decision tree on the resulting dataset.*

Definition 5 *A machine learning **task** describes the objectives of a user regarding their dataset. It can be the training of a predictive model, a regression, a clustering, etc. It will also include the most important performance measures for the user. In the case of a predictive model, as an example, these measures can be the precision of the model, its number of false positives or negatives, or more complex metrics like the area under the ROC curve.*

In general, in the literature, the act of learning from previous *tasks*, in order to recommend possible *workflows* for a new given *task* is called **meta-learning**. This category of recommendation systems is largely described in the survey of Joachim Vanschoren ([Van18]) and divided into multiple categories.

3.2.1 Meta-learning from model evaluation

The most simple meta-learning forms are the **task independent recommendation systems**. Those systems do not run any prior analysis on the new task at hand and only rely on past experiments. In a context where no knowledge of the task at hand is accessible, it is still possible to determine from a database of past experiments, which algorithm tends to perform better in general. The main interest of those task-independent systems is the constitution of portfolios of different data analysis methods and their best global configuration ([Lin10], [Abd18]). These portfolios can be ranked differently depending on the priorities of the user, for instance by preferring a precision-based ranking, or an area under ROC curve ranking ([BSD03],[Dem06]). Moreover, multiple solutions exist to balance evaluation score and training time ([BSD03],[Rij15]). These portfolios are then used to warm start other techniques exploiting the task-specific information, but still relying on general knowledge of algorithms global performance.

One category of methods relying on such previous knowledge is the *relative landmarks* methods, which relies on the performance on a set of pre-selected and well-known models to determine the similarity of two tasks: The meta learner have constituted a portfolio of previous experimentation, containing the performances of different models and their configurations during those experiments. These previous experiments are then compared to the performances of the landmarking models on the task at hand, and it is then theorized that if the landmarks perform similarly on two different tasks, then those two tasks are similar in nature, and the best performing model configuration on the previous task have a better chance to perform well on the new task ([PBG00]). These methods, as the one from [LBV12], often refine the model configuration by warm starting on the best performing model and configuration of the previous task and then proceed to tune the model hyperparameters automatically by relying on further past experiments.

Example 5 *Dr. Kaktus decides to use a data analysis assistance tool to decide which workflow would work best to analyze his dataset. The recommendation system of the tool is based on three landmarkers: decision tree, Naive Bayes, and Nearest Neighbors. Once Dr. Kaktus has entered her dataset in the recommendation system, the three landmarkers are trained automatically on it, and their precision are compared to the results of previous experiments, the matrix of precision is depicted in Table 3.2. The recommendation system*

	Decision tree	Naive Bayes	K nearest neighbors
Dr. Kaktus	0.5	0.9	0.1
Ms. Flour	0.8	0.9	0.7
Pr. Daisy	0.4	0.8	0.15
Mr. Brown	0.7	0.2	0.6

Table 3.2: Matrix of models precision on different experiments

then determines the difference between two experiments as the euclidean distance between the vector of precision of the landmarkers. According to this metric, the closest experiment is the one of Pr. Daisy, which is then used as a basis for the recommendation.

3.2.2 Meta-learning from task properties

Another source of information for the automation of machine learning is the use of metadata obtained from the task and characterizing it. Those metadata are general metrics linked to a dataset, which help to characterize it. There are many types of metadata collected and used by task-centric automation systems, the most commonly used according to [Van18] can be found in Table 3.3. As we can see in this table, these features characterizing the dataset (i.e. *meta features*) can be grouped in different categories. The simplest

ones are the dimensionality meta-features, giving the size of the dataset and the general state of this data. The statistical features are more centered on the characteristics of the different attributes (or features) of the dataset. Information-theoretic meta-features pertains to the classification of the different instances of the dataset and the quality of the information brought by it. The "complexity" meta-features are linked to the difficulty of analyzing the dataset according to different metrics. Finally, the landmarkers and model-based meta-features pertain to the behavior of specific machine learning algorithms when trained on the dataset. The rationale between those meta-features is that if a model behaves similarly on two datasets, those datasets must be similar. These meta-features are then used through different recommendation systems, which will be described in more detail in Chapter 3. In summary, these systems rely on meta-features to compare tasks between them, in order to determine similarities between them. These similarities are then used to select good candidate models for the user's data analysis task, as in [Feu15]. This model recommendation is often accompanied by a hyperparameter optimization, aiming to enhance the performance of the recommended model ([Feu19]).

Other approaches include the building of meta-models: by learning the relationship between a task meta attributes and the efficiency of different models and configurations, it is possible to build a predictive model aiming to rank the models most likely to perform well on the task. There have been numerous works on this domain, as [Bra08] or [LBG15]. Those meta model can also be used to predict other features of a model, as its training and prediction time [Yan18]. This allows the user to specify multiple parameters rather than just the raw precision of the model.

Example 6 *This time, the recommendation system used by Dr. Kaktus is based on meta models. This meta-model has been trained on a large database of past experiments, with a dataset characterizing metadata as an input. The output being the models predicted to perform best on the dataset. Once Dr. Kaktus' dataset has been input in the system, its metadata are passed to the meta-model, which predicts that the models most likely to perform well on it are a Support Vector Machine, a simple neural network, and a J-48 random forest. Dr. Kaktus now only have to choose between those three rather than develop the whole model by herself.*

3.2.3 Description of a workflow recommendation system

Figure 3.1 depicts the global principle of a workflow recommendation as done in [Feu19] or [Ray18]. This process is comprised of three steps :

Category	Name	Utility
Simple	Number of instances	Speed, Scalability
	Number of features	Curse of dimensionality
	Number of classes	Complexity, imbalance
	Number of missing values	Imputation effect
	Number of outliers	Data noisiness
Statistical	Skewness	Feature normality
	Kurtosis	Feature normality
	Correlation	Feature interdependence
	Covariance	Feature interdependence
	Concentration	Degree of discreteness
	Sparsity	inter-class dispersion
	Gravity	Feature redundancy
	Coefficient of variation	Variation in target
	PCA variance	Variance in first principal component
	PCA skewness	Skewness of first principal component
	PCA 95%	Intrisic dimensionality
	Class probability	Class distribution
Information-theoretic	Class entropy	Class imbalance
	Normalised entropy	Feature informativeness
	Mutual information	Feature importance
	Uncertainty coefficient	Feature interdependence
	Equivalent number of feats	Intrisic dimensionality
	Noise-signal ratio	Data noisiness
Complexity	Fisher's discrimination	Separability of classes
	Volume of overlap	Class distribution overlap
	Concept variation	Task complexity
	Data consistency	Data quality
Model-based	Number of nodes/leaves	Concept complexity
	Branch length	Concept complexity
	Nodes per features	Feature importance
	Leaves per class	Class complexity
	Leaves per agreement	Class separability
	Information gain	Feature importance
Landmarkers	Landmarker k-NN	Data sparsity
	Landmarker Tree	Data separability
	Landmarker Linear	Linear separability
	Landmarker NB	Feature independance
	Relative landmarker	Probing performance

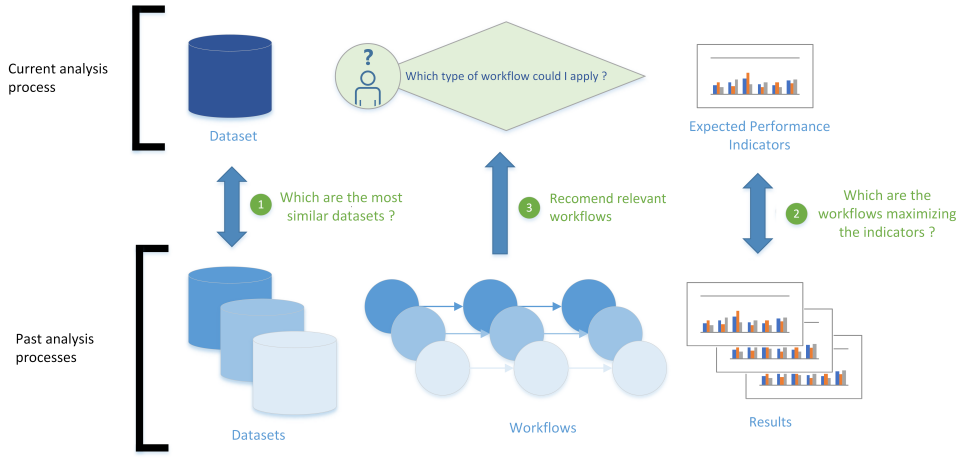


Figure 3.1: Global steps of a recommendation process according to [Ray18]

1. The system finds a set of datasets similar to the user's dataset in his meta-database of past data analysis experiments.

2. Among those similar datasets, the system rank the workflows used when performing the past experiments according to their performance indicators.

3. The workflow performing best on a dataset similar to the user's dataset is then recommended to the user.

Example 7 *Dr. Flour is a flower biologist researching the different species of plants she has; namely iris setosa, iris virginica, and iris versicolor. She's trying to come up with an efficient model of classification given, among other criteria, the petal and the sepal size. Now, Dr. Flour has no real expertise in the field of data classification, and would rather reuse processes designed by experts. Dr. Flour primarily wants her model to be the most accurate (considered as the only performance indicator).*

At the same time, Dr. Kaktus is a famous geneticist who came up with a computer model that can classify cacti flowers based on their size. Fortunately, Dr. Kaktus uploaded his data and his analysis workflows on a machine-learning platform (such as OpenML [Van13]) offering to anyone to evaluate or run operations on it.

The objective of the recommender system is then to be able to determine that the data and objective of Dr. Flour are reasonably close to those of Dr. Kaktus, and thus to reuse the analysis workflows built by Dr. Kaktus to solve Dr. Flour problem.

The hypothesis here, backed by the whole field of meta-learning ([GVB04]), is that a data analysis workflow that performed well on a given dataset has a good chance to perform well on a new dataset if it is *similar* to the first.

In order to determine the similarity of two datasets, many recommendation systems rely on a similarity measure.

3.2.4 Determining the (dis)similarity between two datasets

The auto-SKlearn and William Raynaut's systems ([Feu19], [Ray18]) both use their measure as a *dissimilarity*, which allows them to use it as a *distance*. This process is more intuitive to use than a *similarity* measure, as it is often simpler to count differences rather than similarities. According to William Raynaut ([RSV17]), those dissimilarity measures have a set of necessary properties :

Definition 6 *Let A be a set and d a function* : $A^2 \to \mathbb{R}$.
d is a ***dissimilarity function*** *on A if and only if, $\forall x, x' \in A$:*

- $d(x, x') \geq 0$ *(Positivity)*
- $x = x' \to d(x, x') = 0$ *(Indiscernibility of identicals)*
- $d(x, x') = d(x', x)$ *(Symmetry)*

Those properties are the same as the ones of a *distance* when defined on a multidimensional space, except for the absence of the identity of the indiscernibles (the inverse of the second property). This absence is caused by the approach of comparing datasets.

A dataset, by nature, can have the order of its columns and lines changed without consequences for its relevances, as long as the relations between instances and attributes is respected. In short, the order in which the instances are presented is not relevant, and neither is the order of the attributes.

Thus, in order to compare different datasets, it is not possible to rely on the raw data, as its order can be changed. Instead, the comparison algorithms relies on *metadata*. Most of the literature ([Feu19; Feu15; NHK14]) rely on the dataset *metadata* alone. A lot of those are the criterion described earlier in Section 3.2.2, through Figure 3.3. Although, when comparing datasets, the authors of [Ray18] decided not only to use this metadata but also *attributes specific metadata*. This metadata consists of criteria similar to those of the dataset but is calculated only on the attributes. As an example, those metadata can be the mean value of the attribute, if it is numeric or nominal or its variance. A full list of the criterion used by [Ray18] is available in their thesis.

In order to conjugate the differences between the attributes, the authors propose to form pairs of attributes as depicted in Figure 3.2. The goal of those pairs is to minimize the

total dissimilarity of each pair of attributes. This is achieved by using a greedy algorithm to form pairs of attributes. In the case where the two compared datasets do not have the same number of attributes, the supernumerary attributes are paired with a theoretical void attribute. Once this *attribute specific* dissimilarity is calculated, it is then combined with a more classical *dataset based* dissimilarity.

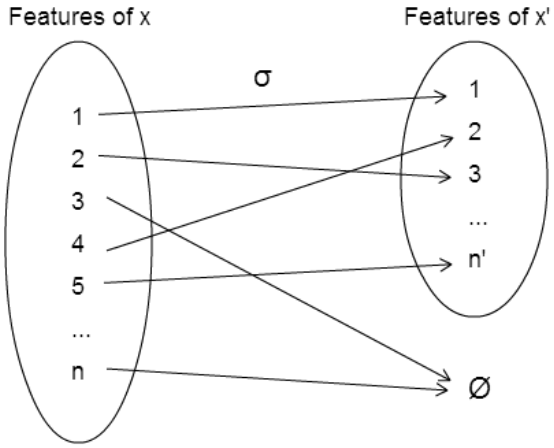


Figure 3.2: pairing the attributes (features) of two datasets

Experiences conducted in [Ray18] have shown that this finer dissimilarity measure is more accurate than other measures of the state of the art.

3.3 Improving collaborative filtering by using context

Traditional collaborative filtering approaches base their recommendations on similarities between users' problems, identifying intrinsic traits they have in common. A commonplace improvement of those systems is to take into account the *context* of the user.

Context-aware recommender systems should be preferred when the context of the user is complex and more prominent [AT08]. In this case, the information obtained from multiple contexts can be very useful to improve the relevance and effectiveness of the recommender systems. Such approaches have attracted particular attention over the last few years. For example, [IY17] shows that detecting user emotion (context) and factoring it into a collaborative filtering approach increased user satisfaction. [Wan17] proposes a system suggesting collaborations between universities and industries based on the identification of similar contexts of researchers (defined on a multitude of aspects). [ZMB15] develops a similarity-based context-aware approach under the assumption that recommendations should be similar if the contextual situations of the users are similar. They demonstrated that integrating a similarity measure between multidimensional contexts could improve

precision scores. [AT11] describes in detail the existing recommender system methods and proposes solutions to take into account the user's context in addition to the user himself.

Three main methods are highlighted in [AT11]. For each of these methods, we illustrate it with an example based on a movie recommendation system. We consider a user who is searching for an action movie. The system has access to the previous movies he watched and which he liked, as for a classical collaborative filtering approach. The system also has access to the context of our user: apart from movies, he is interested in video games and sports, but doesn't like politics and traveling.

- **Method 1: Pre-filtering.** In this method, the context is taken into account when considering the already available data. The context is then considered as additional data in the recommendation task. The main question then becomes the weight we have to allow to this data, depending on the importance of the user's context in the query. As an example, a biologist looking for interesting data could benefit in a greater importance of his own context in the recommendation: his precise field of work is not always apparent in his activity, along with his current center of interest.

  In our example, the recommendation system would find people who watched and liked the same movies as our user, to know what these people also liked. Then, among those people, the ones who like video games and sports but dislike politics would be given a greater weight before running the recommendation algorithm. This makes the system base its recommendation on people with tastes even more similar than those of the user.

- **Method 2: Post-filtering.** In this case, the context is taken into account once the recommendation has been made. As an example, if the basic recommendation system has recommended ten items of interest for the user, these items will then be ranked according to the user's context. This reduces greatly the importance of the context in the recommendation, without completely eliminating it, as the user is still more likely to get content relevant to his context.

  For our example, the system first proceed as a classical collaborative filtering recommendation system by basing its recommendation on the persons who have watched and liked the same movies as our user. Then, the system produces 15 possible movies that could interest the user. These movies are then ranked through the context of the user: among these 15 movies, the movies presented first are the ones best liked by other persons who are interested in video games and sports but don't like politics.

- **Method 3: Model-based filtering.** This application is relevant to model-based recommendation systems. Here, the context is considered as a new dimension when creating the recommendation model space. The end ranking of the recommendation can then be generated using a Pareto front or a weighted manhattan distance. This allows to use of the context in the recommendation without intertwining it with the user's data: the user and his context are still considered as separate entities in the recommendation system but are used conjointly for the recommendation.

  In our example, the tastes of the user outside of movies would be integrated into the recommendation system, establishing two rankings: The first is the ranking of the movies liked the most by people who watched and liked the same movies as the user (ranking A). The second is the ranking of the movies liked the most by people who like video games and sports and dislikes politics (ranking B). The final recommended movies would be the ones performing best in both rankings. As an example, a movie second in ranking A and fourth in the ranking B would be recommended over a movie first in ranking A but 100^{th} *in the ranking B.*

There exist solutions in order to use combinations of these methods, notably by using multiple context points in different stages of the recommendation (e.g. [Ado05]). Adomavicius cites an example in [AT11] by saying that different parts of the user's context could be used in different steps of the methods. Moreover, it is possible to use multiple context data in the same method. A recommendation method could then use a group of context elements in pre-filtering, others in its own model, and a final group as post-filtering. In the example of movie recommendation, there is a variety of pertinent context elements: the time at which the user wants to watch a movie, his tastes outside of movies, if he is watching the movie with someone or not, etc. Those different pieces of information may be used during different steps of the recommendation, rather than using them all at the same time.

Having collected methods of collaborative filtering recommendation, it is now necessary to see their possible applications in machine learning.

3.4 A confidence problem

In general, machine learning recommendation systems offer very accurate analysis workflows and predictive models (drawing strength from workflows performed by past users). Yet, the interaction with these systems is often limited to execute the predictive model

proposed, without an easy way to validate and personalize it. This is a major drawback by which a user can lose confidence, due to a lack of explanations in the recommender system results. Indeed, neophyte users tend to struggle giving credence to a system they do not understand and are not familiar with. Given the fact that important decisions can be made using such a system, giving the user an opportunity to have confidence in the system is important. For example, the importance of transparency has been recognized for a long time in expert systems, as in [BS85], and studied more widely in the recommendation context in [TM15b].

[WMR17] and [RSG16] explains that a recommended content is at risk of being underused if the product of the recommendation is not understood by the user. This is combined with the "black box" aspect of a model, in which internal code and working cannot be accessed.

The paper [WMR17] lists a lot of the practical and legal implications of this black box state. A user who cannot fully understand how the model works would be right to distrust this model when applied to sensitive domains, as it can rely on illegal data or end up causing discrimination in the decisions taken by the model. As an example cited by the article, an American bank using a predictive model to decide whether or not to allow a loan to a client has been sued over the decisions of this model. Indeed, as the data used for its training was biased toward the ethnic profile of their clients, the model learned this behavior and reproduced it, even if the ethnicity of the client was not a part of the original dataset. Indeed, the areas of birth and other social and living factors are strong indicators of the possible ethnicity of a person, which ended up biasing the model against peoples from regions and sociocultural origins mainly containing people of color. Thus, as illustrated in this example, relying solely on the expertise of a recommendation system can harm many persons if it is used without understanding its inner working.

To overcome this *black box* problem in the recommendations, several solutions exist in the literature, whose review can be found in [TM15a]. Particularly, a number of goals characterizing the different types of explanations in recommender systems is proposed, and how to evaluate them. In particular, the notions of

- *Transparency* (i.e. how the system works as in [Cra08]),
- *Trust* (i.e. perceived confidence with the recommendations as in [CP05]),
- *Persuasiveness* (i.e. user acceptance with the recommendations as in [Cra08]),
- *Effectiveness* (i.e. make better decisions as in [Sha13]),

seem to fit at the best the objectives of recommendation explanations proposed in our framework.

3.5 Synthesis

A large number of workflow recommendation systems rely on learning from past experiments. Those past experiments serve as a basis for meta-learning, which tries to find experiments similar to the user's in order to recommend workflows. Because of this context, a majority of the workflow recommendation systems can be assimilated to *collaborative filtering* approaches, as they rely on previous users' activities to recommend content to new users. In general, the domain of data analysis processes automation has been largely explored recently, and a lot of possible recommendation systems are available depending on the task at hand.

Most of those systems rely on a dissimilarity measure in order to quantify the differences between two datasets, often by comparing their metadata. The work of [Ray18] improves those systems by taking more information into account when calculating the dissimilarity. Yet, an important amelioration in a lot of collaborative filtering systems is taking into account the user's context. Despite this, this amelioration is rarely present in the domain of workflow recommendation systems based on collaborative filtering.

Importantly, as depicted in 3.4, we can see a granularity in these recommendation systems. Some of them recommend a full workflow, from data pre-treatment to model training, including its hyperparameters and post-treatments ([Feu19]). Some others, instead, prefer guiding the user through the construction of a workflow by recommending different steps depending on the problem and the previous steps already in place ([Dem13]).

Most of the tools available to the public and aimed at non-experts are often a combination of diverse recommendations and automation tools. This is done in order to automatize the machine learning process as much as possible, thus eliminating the need for human participation.

There is an important caveat in this approach: by creating a tool the user does not understand, the process will diminish the likelihood this tool will be used to its full potential, due to a confidence crisis. This lack of confidence phenomenon has been studied and detailed in [RSG16] and [WMR17], which describe the need for a minimum of comprehension of a model to be used efficiently and safely. From this phenomenon arises a need for guidance in the creation of machine learning models without the elimination of the user's participation. By getting the user involved, these tools would achieve more consistent use

of their results.

Recommendation system	Meta learning type	Category	Coverage
[Feu19]	Task properties	Similarity based	Workflow + hyperparameters
[Ray18]	Task properties	Similarity based	Workflow
[LBV12]	Model evaluation	Relative landmarks	Hyperparameters
[Abd18]	Model evaluation	Task independant	Workflow
[Feu15]	Task properties	Similarity based	Workflow
[Dem13]	Model evaluation	Meta model	Workflow steps
[Bra08]	Task properties	Meta model	Workflow + hyperparameters

Table 3.4: Summary of different existing recommendation methods for machine learning.

I didn't want to just know the names of things. I remember really wanting to know how it all worked.

Elizabeth Blackburn

I disagree strongly with whatever work this quote is attached to.

Randal Munroe, xkcd

Chapter 4

Prediction explanation

The existing systems and toolkits for machine learning (ML) and data analysis in general mostly focus on recommending a model and explaining the principal characteristics of the data analyzed. This approach has proven particularly helpful for expert users, but still requires advanced knowledge of data analysis. Indeed, some of the most well-known data analysis platforms such as Weka [Hal09] or Knime [Ber07] provide detailed descriptions of the methods and algorithms they include, often giving usage examples. Unfortunately, detailed descriptions are not enough to understand this kind of data without actual training in data analysis.

As we have seen in Chapter 3, there exists a need for guidance in understanding predictive models. Hence the importance of one of the identified issues: **How can we help a non-expert understand and analyze the results produced by a data analysis process?**

The main deterrent to the comprehension of the majority of machine learning models is their "black box" aspect: once a "black box" model has been trained, it is not possible to know the exact reasoning behind the classifications performed. Of course, some models remain interpretable by nature, as they are simple enough to be understood when looked at code-wise. As an example, a decision tree can always be represented as a tree, which will be interpretable for a human, unless the number of branches becomes too large to be practical. Another example of an interpretable model is the linear regression of a dataset. As a linear regression consists of a simple linear equation, a human can understand its working. The "black box" problem arises when more complex models are used. For those cases, the information necessary to understand directly the model becomes too large to be encompassed for a human. As an extreme example, we can always represent a neural network as a lattice of neurons, but the representation of thousands of nodes and paths

would not be useful for a human observer.

Thus, we cannot directly access the information on the internal working of a complex model. Yet, it is possible to observe the *effects* of this working, namely, the predictions done by those models. We can consider each prediction made by a model as an additional clue on the way the model functions internally. That is the approach used by a large part of the literature to understand how a model works. In those approaches, two main goals can be seen: *explaining the model behavior globally*, in order to understand it in the general context, or *explaining the model behavior for a particular prediction.*

In this chapter, we first see how models are explained through their predictions in Section 4.1. Through this section, we first take a quick glance at *global* explanations, before focusing on *single prediction* explanation. Then, we focus even more on *additive* explanations in Section 4.2. Finally, in Section 4.3 we look at *how* predictions explanations can be used to help users.

4.1 How to explain a model by exploiting its predictions

Explaining the influence of each attribute of a dataset on the output of a predictive model has been explored largely. A few of the works pertaining to global attribute importance on a model can be seen in these papers: [Alt10] [KR92]. The most recent methods are based on swapping the values of attributes in the dataset and analyzing which swaps affect the trained model predictions the most. The more modifying the values of the attributes affect the predictions, the more this attribute is considered important for the model, as a whole. These methods are often used during feature selection, allowing to opt-out attributes not useful for the model.

Another approach for understanding a model is described by Helenius et al. in [HPU17]. In order to understand how the studied model works, they seek to understand which attributes of the datasets are "linked" to each other, according to the model. In order to do so, they proceed by randomizing the values of potential groups of attributes. When randomizing values inside a group of attributes is making the predictions of a model vary more than a predetermined threshold, they consider the attributes as being linked together. This process creates a grouping of the dataset attributes, which inform the users on how they are interacting with each other according to the model.

The problem with a fixed global influence is that predictive models are often not consistent with the whole dataset on which they are trained. These global influences give us an insight into the general working of the studied model, but there will often be particular

regions of the dataset where the model will deviate from this general image. Thus, there also exists a need for a single instance prediction explanation, showing the user how a particular instance has been classified, independently to the rest of the dataset. These methods aim to provide insight into the global influence of each attribute, rather than on their influence on a single prediction.

These global influence methods have served as a basis to [CMB18] which use the same randomization technique but on the scale of a single prediction. Given a single instance, the importance of each attribute is obtained by looking at the evolution of the prediction performance of the model on the instance when all of its values are swapped with other values of the dataset except the value of the attribute being studied. The more the prediction varies with the values swapping, the less the fixed attribute is important for the prediction. This method has the interest of relying on the same principle as the global technique which is largely recognized, but the main caveat of this is the computational cost needed for explaining the prediction of only one instance, as a large number of new predictions has to be generated for each single instance prediction we want to explain. Another caveat for us is that this prediction explanation is realized from the point of view of model performance. Meaning that their metric shows which feature improves the performance of the model, rather than which feature the model consider as important for its prediction. **If this line of reasoning is really interesting for the model explanation field, it does not correspond completely to our scope, as we are aiming to help users understand how a model works, and not how to improve it.**

Another approach can be found in [WMR17], which inspired many of the functions of [Wex19]. The principle here is to determine the smallest change needed in order to change the classification of an instance by the model. It has been integrated into the What-if tool, as depicted in figure 4.1. This tool permits the user to select a particular data point in the dataset, and visualize the nearest point classified differently. This is displayed along with the differences between the two points, thus highlighting what let the two points to be classified differently.

Example 8 *This method is often used when studying and training neural networks, notably in the context of adversarial networks. Adversarial networks are two neural networks trained in opposition. The first one is trained to perform the wanted task, as classifying an image, while the other is trained to create instances designed to fool the first network. In the example of image classification, the second network will perform a modification as little as possible in order to change the classification of the object. This modification is often*

not perceivable by the human eye but can change completely the nature of the recognized object. These approaches led to the development of objects like the ones depicted in figure 4.2, which are glasses modifying enough the face of the wearer to change completely the person being detected. Those glasses have been created by [Sha16].

These methods give us an insight into how the model works, as they display the attributes which could put the instance in another class if their value was changed slightly. However, if these methods are interesting for analyzing the important points of a model, this kind of information would be far less useful to a non-expert, as they already require the user to understand the importance of this information and draw conclusions on it by himself.

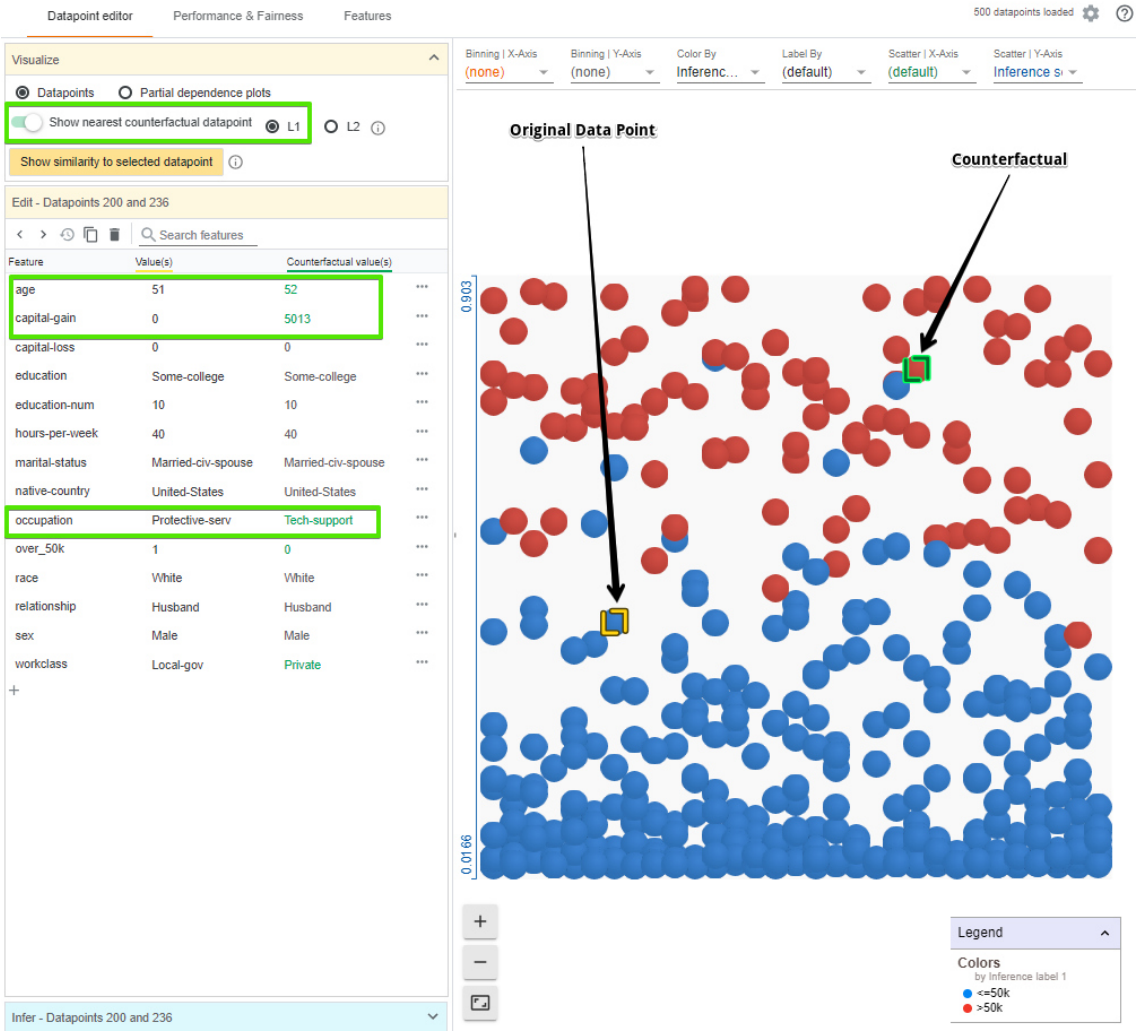


Figure 4.1: The What-if tool allows the user to see the nearest point in the dataset classified differently than the one selected. On the left is highlighted the main differences of the two points, which led them to be classified differently.

One of the early explanation methods can be found in [ŠK08b]. In this explanation method, the weight of an attribute on the prediction is estimated by seeing the difference in the prediction of the model with and without the attribute. This absence of an attribute is

Figure 4.2: Adversarial glasses which fools facial recognition neural networks into classifying the wearer (top image) as another person (bottom image) [Sha16]

simulated by a weighted mean of the predictions of the model with all the possible values of the attribute, weighted by their probability of appearing in the dataset. This is faster than the method of [CMB18] as only one value is randomized. Later, in [SK10], the possibility of retraining the model entirely without the considered attribute in the dataset is proposed, which consists of an interesting trade-off: the time needed for retraining a model for each attribute of a dataset can be considerable, but once this training has been done, the prediction comparison can become near-instantaneous. These methods are named *additive methods* by Lundberg et al. ([LL17a]), who regrouped a large number of similar methods of explanation.

4.2 Additive explanations

Indeed, a great number of works pertaining to prediction explanation led to [LL17a], which theorized a category of explanation methods, named *additive* methods, and produced an interesting review of the different methods developed in this category. Some of these methods are described in detail in [DSZ16] and [SGK17]. They are summarized in [LL17a] as methods attributing for a given prediction a weight to each attribute of the dataset.

This creates a very simple "predictive model", mimicking the original model behavior locally. Thus, we have a simple interpretable linear model which gives information on the original model inner working in a small vicinity of the predicted instance. The methods, from which these weights are affected to each attribute, vary between the different *additive*

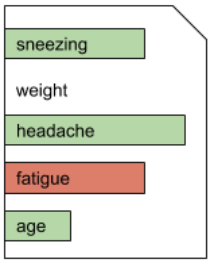


Figure 4.3: An example of additive explanation, displayed as colored weight bars. Green for positive influence, red for negative.

methods, but the end result is always this vector of weights.

For instance, [ElS19] proposes an intuitive method for explaining the prediction of a model on a single instance. First, new random data points are generated in the close vicinity of the studied prediction. The analyzed model is then asked to perform predictions for those points. Hence, a "sub dataset", comprised of those new points and their affixed predictions. A linear model is then trained on this sub dataset: thus, the linear model mimics the behavior of the original model in the vicinity of the explained data point. As a linear model is simply a set of weights and conditions on each attribute of the dataset, it is interpretable for a human, and can be used as a description of the original model behavior for the prediction on the data point.

Another method is the one described in [SK10]. In this article, they describe a method which relies on comparing the differences of the predictions of the model in different configurations. In this method, the importance of an attribute is considered as the difference in the prediction of the model when the attribute is added to the dataset. Thus, the weight assigned to an attribute will be dependent on the information brought by this attribute to the prediction.

Example 9 *A physician wants to predict if his patients have the flu or not by using a predictive model. In order to make him understand the predictions better, a prediction explanation is displayed to him for each prediction. For instance, a patient is predicted to have the flu with a 75% confidence by the model, and the doctor is displayed a prediction explanation as in figure 4.3. For this patient, his values for the attributes sneezing and headache have been determined to be indicative of the flu by the model, while the value for fatigue has reduced the certainty of the flu according to the model. Finally, the user can see that the attributes age and weight were not used much by the model for this prediction.*

[LL17a] highlights several interesting properties about these methods :

- Local precision: The system describes precisely the model in the close vicinity of the explained instance.
- "Missingness": If an attribute is missing for the prediction, the method does not give it a weight, or gives it a weight of zero.
- Consistency: If the explained model changes in a way that makes an attribute more important, or does not change its importance, its attributed weight is not diminished. This property is important, as some of the early prediction explanation methods could have an erratic behavior in some cases, as shown in an example of [LL17a].

This type of prediction explanation is quite interesting, as we are aiming to facilitate the understanding of any machine learning models for users without particular knowledge on data analysis or machine learning. Thus, it is more relevant to focus on the works as [SK10] or [DSZ16], cited as *additive* methods, as they generate a simple set of importance weights for each attribute. This set of weights is easy to interpret, even for someone without expertise in machine learning. Yet, these methods have a major deterrent: their complexity makes them difficult to use for the average user. That is why [LL17a] explored methods to generate explanations faster, but at the cost of very restricting hypotheses, as the Independence of each attribute of the dataset, or the linearity of the model, which is not always the case. The ability to explain the prediction of any model thus appears to be a key point for allowing a broader public (non-expert) to access and use machine learning models. This need led us to consider the diverse explanation systems, developed in the literature, as having a major interest in giving more autonomy to domain experts performing data analysis tasks. Yet, the computational load found in the most generic methods can be a hindrance to their use. In this paper, we seek to select a prediction explanation method as generic as possible and try lowering its computing time without losing too much information.

4.3 Applications of single prediction explanation

The possible applications of prediction explanations have been investigated by [RSG16]. According to their paper, the interest in explaining a predictive model is threefold:

- First, it can be seen as a means to understand how a model works in general, by peering at how it behaves in diverse points of the instance space.
- Second, it can help a non-expert user to judge the quality of a prediction and even

pinpoint the cause of flaws in its classification. Correcting them would then lead the user to perform some intuitive feature engineering operations.

- Third, it can allow the user to decide the type of model preferable to another one, even if he has no knowledge of the principles underlying each of them.

Some of these applications are illustrated in example 10, which relies on Figure 4.4 :

Example 10 *A physician has trained a model to predict the likelihood of his patients to have the flu. In the first instance, the prediction is very uncertain, and the user cannot really decide on this prediction. In the second instance, a prediction explanation is given to the physician: he can see that the sneezing and headache of his patient have increased the likelihood of the flu according to the model, but the absence of fatigue has reduced it. This explanation can effectively explain why a result has been obtained, helping a user to make an informed decision.*

Another application is described in example 11, which is illustrated by Figure 4.5

Example 11 *a doctor has to choose between two potential models trained for predicting diabetes in his patients. This decision can be greatly aided if he has access to the influence of the different attributes on each prediction. Here, it is clear that for the particular patient displayed in Figure 4.5, model A predicts a possibility of diabetes because of his diet, but that his regular exercise reduced it. The model B don't use exercise as much as the other, but rather uses the weight of the patient in addition to his diet. Given this fact, the doctor can make a decision on which model is the best according to his own knowledge of diabetes and its underlying causes.*

4.4 Synthesis

When trying to explain a predictive model, it is often not possible to access its internal working directly. Instead, it becomes necessary to understand it through its outputs produced and its interaction with the data analyzed .

Thus a lot of different approaches have emerged throughout the works in the literature. The differences not only relies on *what* is being analyzed, but also *how* this analysis is obtained. Some papers illustrate how a model functions through the minimal changes needed to modify the result of a prediction ([WMR17]), thus the relations between data and prediction are observed. Others prefer observing what happens when the values of

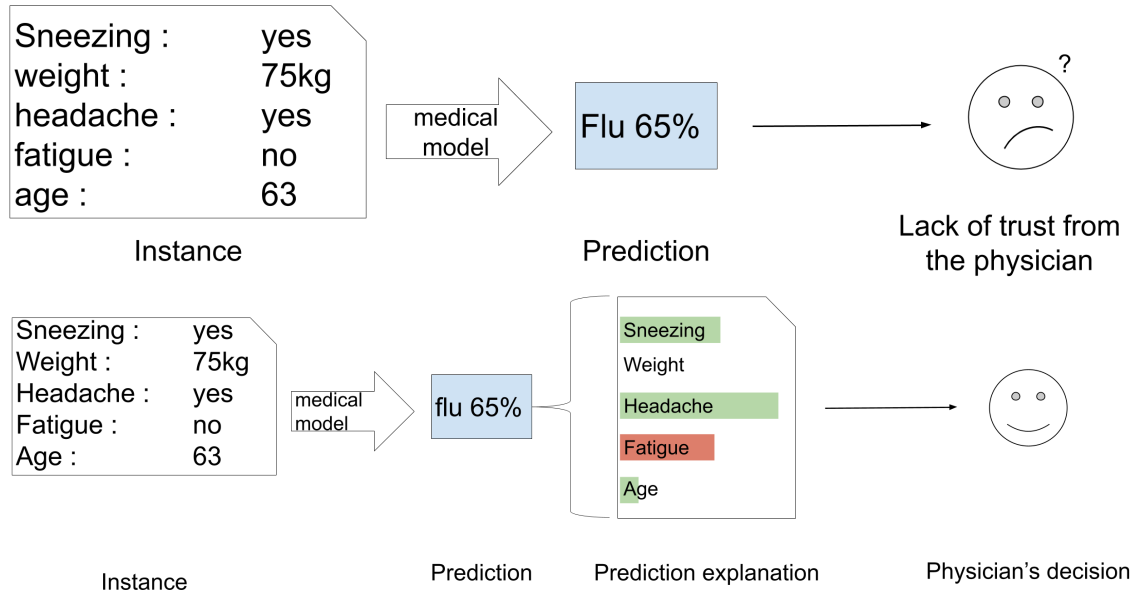


Figure 4.4: An example of prediction explanation and how it can help an user non expert in machine learning to make a decision

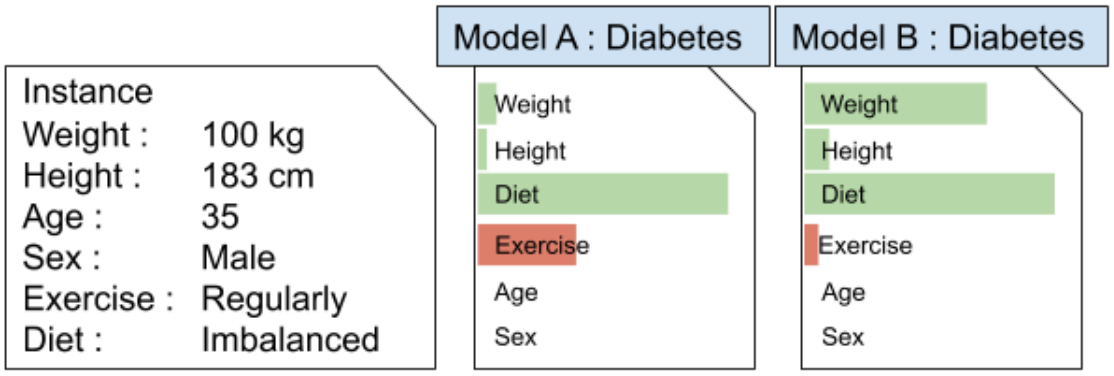


Figure 4.5: Comparing two models on the same dataset via prediction explanation : model A relies more on exercise (which votes against the current classification), while model B relies more on weight

	Explanation format	Observed metric	Accessibility	Quantity of information	Calculation speed	Accuracy
Counterfactual explanation	Closest different classification	Prediction variation	Hard to interpret	High	Fast	High
Attributes grouping	Groups of interacting attributes	Prediction variation	Medium	Medium	Medium	High
Fast additive explanation	Weights vector for each attribute	Model performance/ Prediction variation	Very accessible	Medium	Fast	Medium
Complete additive explanation	Weights vector for each attribute	Model performance/ Prediction variation	Very accessible	Medium	Slow	High

Table 4.1: The different explanation systems presented in this chapter, with their pros and cons.

different attributes are randomized ([CMB18]), but among this category of explanations methods, there are divergences on which results are being observed. In some cases, the measure for the importance of the change will be the change in the precision of the model, while other approaches will prefer observing the changes in the predictions of the model, as an example.

Among all those different paths to explore, the simplest way to represent how a model works might be the *additive* explanations as they simply produce a vector of weights, each weight representing the importance of an attribute for the model. But even among those, a lot of different approaches exist. Some methods rely on observing what happens when an attribute is removed, others do the opposite by watching what happens when an attribute is added. Other methods also rely on training a linear model on a very small subset of data points generated by the original model.

We can see in Table 4.1 that this interpretability of the additive explanation comes either at the cost of a long calculation time or a loss in the accuracy of the explanation. On another hand, the counterfactual explanation can be very informative and is fast to compute, but it will be hard for a beginner in data analysis to interpret this information. Finally, the attribute grouping approach to explaining a model is limited as it is only applicable on a global scale and might be lacking in information for the user, but it is still an interesting approach, especially if it can be used in conjunction with the other approaches.

Even with such a large array of possible choices, a large gap is left in the literature. There exist a lot of ways to explain a model, but there are almost no researches on *how* to use those explanations. Some quick suggestions have been made by Ribeiro in [RSG16], but those suggestions have not been used or tested.

Conclusion and positioning

Our analysis of the state of the art revealed that the majority of the systems aiming to facilitate data analysis for the user are either based on :

- A partial or complete automation of the data analysis process;
- The providing of visual assistance aiming to facilitate the understanding of the studied data;
- A visual workflow construction interface making the building of a data analysis process more approachable.

Throughout the course of this thesis, our ambition is to help a user to do complex data analysis tasks, without needing extended expertise in this domain. Thus, the first solution would seem to be the best fit for this goal. However, it has been demonstrated in multiple works and papers that a lack of comprehension of a data analysis model can cause it to be harmful in multiple ways. Namely :

- Under-usage of the resulting model ([RSG16]);
- Lack of trust in the produced results and predictions ([RSG16]);
- Misuse of the produced results ([WMR17]);
- Reproduction of biased or harmful behaviors ([WMR17]).

These major drawbacks lead us to consider the full automation of a data analysis process as useful for speeding up simple applications done by an expert, but as not ideal for a non-expert user. For those kinds of users, the usage of automation is required in order to make decisions they cannot make in an informed way, but the automation must be accompanied with guidance in its usage, and include the user in the data analysis process. This gives more confidence to the user in the produced results, and help him to make sure the model corresponds to his needs.

In order to achieve a workflow recommendation process useful for a non-expert user, we need to rely on elements he is capable of providing, without asking him too many technical details, except in his own domain of expertise. With this goal in mind, context-aware recommendation systems is a promising lead. As the user is an expert in his own domain, but not in data analysis, he knows *what* he wants to analyze, but do not know *how* to do it. As such, by considering the user's data as his *context*, we can use what the user knows in order to perform recommendation even if he lacks data analysis expertise.

Finally, in order to overcome the *black box* problem cited in many articles in the literature, we need to make the recommended workflows interpretable by the user. For this, **it becomes important for the user to be able to explore his model, not only in a global way but also on an instance-by-instance basis.** This exploration brings much more insight to the user on the model inner workings while providing important information on the reasons for a particular prediction.

Many of the explanation tools rely on the user having minimal knowledge of data analysis, but *additive* explanations just assign a weight to each attribute of the user's dataset. This is a very good solution, as the user knows his own data, and can understand this relation between his data and the model. By assigning these weights, *additive* explanations

put the predictions of the models in the context of the user's data, allowing him to easily understand this explanation. Although, the computational weight of the more developed additive method may become a deterrent for their usage in a framework. Thus, we take on this problem in chapter 6, in order to create an explanation balancing accuracy and speed.

By using these *additive* explanation, we build an environment allowing the user to explore the possible models recommended, and act on this exploration by making real model building decisions.

In the following part of this thesis, we describe our work and propositions in order to answer the lack highlighted during this state of the art.

References

[Alt10] André Altmann et al. "Permutation importance: a corrected feature importance measure". In: *Bioinformatics* 26.10 (2010), pp. 1340–1347.

[Ber07] Michael R. Berthold et al. "KNIME: The Konstanz Information Miner". In: *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. ISSN: 1431-8814. Springer, 2007.

[CMB18] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. "Visualizing the feature importance for black box models". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2018, pp. 655–670.

[DSZ16] A. Datta, S. Sen, and Y. Zick. "Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems". In: *2016 IEEE Symposium on Security and Privacy (SP)*. ISSN: 2375-1207. May 2016, pp. 598–617.

[ElS19] Radwa ElShawi et al. "ILIME: Local and Global Interpretable Model-Agnostic Explainer of Black-Box Decision". In: *European Conference on Advances in Databases and Information Systems*. Springer. 2019, pp. 53–68.

[Hal09] Mark Hall et al. "The WEKA data mining software: an update". In: *ACM SIGKDD explorations newsletter* 11.1 (2009). Publisher: ACM, pp. 10–18.

[HPU17] Andreas Henelius, Kai Puolamäki, and Antti Ukkonen. "Interpreting classifiers through attribute interactions in datasets". In: *arXiv preprint arXiv:1707.07576* (2017).

[KR92] Kenji Kira and Larry A Rendell. “A practical approach to feature selection”. In: *Machine Learning Proceedings 1992*. Elsevier, 1992, pp. 249–256.

[LL17a] Scott M Lundberg and Su-In Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4765–4774.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “"Why Should I Trust You?": Explaining the Predictions of Any Classifier”. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. event-place: San Francisco, California, USA. New York, NY, USA: ACM, 2016, pp. 1135–1144.

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning Important Features Through Propagating Activation Differences”. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML’17. event-place: Sydney, NSW, Australia. 2017, pp. 3145–3153.

[Sha16] Mahmood Sharif et al. “Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition”. In: *Proceedings of the 2016 acm sigsac conference on computer and communications security*. 2016, pp. 1528–1540.

[ŠK08b] Erik Štrumbelj and Igor Kononenko. “Towards a model independent method for explaining classification for individual instances”. In: *International Conference on Data Warehousing and Knowledge Discovery*. Springer. 2008, pp. 273–282.

[SK10] Erik Strumbelj and Igor Kononenko. “An Efficient Explanation of Individual Classifications Using Game Theory”. In: *J. Mach. Learn. Res.* 11 (Mar. 2010). Publisher: JMLR.org, pp. 1–18.

[Wex19] James Wexler et al. “The what-if tool: Interactive probing of machine learning models”. In: *IEEE transactions on visualization and computer graphics* 26.1 (2019), pp. 56–65.

[WMR17] Sandra Wachter, Brent Mittelstadt, and Chris Russell. “Counterfactual explanations without opening the black box: Automated decisions and the GDPR”. In: *Harv. JL & Tech.* 31 (2017), p. 841.

Part II

Building a system for assistance in data analysis : Contributions

Preamble

During our exploration of the state of the art, we have analyzed the solutions already existing for tackling the three issues we decided to address. We also have identified the lacks of the literature regarding those issues. In this second part, we describe our propositions for overcoming those lacks.

Concerning our issue of **efficiently recommend a data analysis workflow to a non-expert user**, we have seen that a lot of collaborative filtering recommendation systems are used for recommending machine learning workflows, but that those systems are rarely using the user's context for their recommendation. Moreover, we have seen that a recommendation system might suffer from a distrust problem because of its opacity. Thus, in our contribution, we propose a novel recommendation method, aiming to put the user back in the process. This is achieved by asking the preferences of the user in terms of performance for the final model, and taking those preferences into account when recommending a possible workflow. This proposition is described in Chapter 5.

But, we also have seen that a predictive model needs an explanation to become more effective when used by a non-expert user. Thus, if we want to address our issue of **helping a non-expert understand and analyze the results produced by a data analysis process**, we have to consider the current lacks in the domain of prediction explanation. As we have seen, explaining the functionality of a model on a global scale have been largely explored, but difficulties remain when explaining single predictions. Those explanation systems struggle to be applicable for all types of models while remaining accurate without needing a long time to calculate. Thus, in Chapter 6, we propose our method for explaining the predictions of a model, by automatically detecting the pertinent attribute in the prediction, and thus optimizing the calculation of its explanation. We then experiment with this method. Those tests show that our proposition achieves a satisfactory level of precision, without costing too much calculation time while being applicable for all types of predictive models.

Finally, we have seen that tools aiming to assist users in machine learning often tend to be oriented toward users expert in data analysis. As we aim to **help "non-expert" users to build their own data analysis model**, we want to create a more approachable environment. Thus in Chapter 7, we describe a novel framework for assistance in machine learning. This framework uses our previous proposals in an original way to guide a non-expert user through the process of creating a machine learning model. Our proposal aims to exploit the user's field knowledge to guide the machine learning training process and to

explain this process to the user. With it, we want to make the whole process understandable for the user, and give him confidence in what is being done.

Let me do it. You tell me when you want it and where you want it to land, and I'll do it backward and tell you when to take off.

Katherine Johnson, on the trajectory of the first Apollo mission

Humans are allergic to change. They love to say, 'We've always done it this way.' I try to fight that. That's why I have a clock on my wall that runs counter-clockwise.

Grace Murray Hopper

Chapter 5

Putting the user back in a recommendation system

5.1 Introduction

Our goal in this chapter is to establish our proposal for a collaborative filtering recommendation system, which acts as a basis for our framework, and later, our prototype.

As we have seen in our exploration of the state of the art, in Chapter 3, collaborative filtering approaches are largely considered as the most effective when recommending machine learning workflows. Thus, our proposition will also be based on a collaborative filtering system.

But, as we also have seen in our state of the art, approaches from the literature do not use context when recommending a machine learning workflow. The problem here is that the issue we want to address is to **efficiently recommend a data analysis workflow to a non-expert user**. Thus, we want to create a solution-oriented toward non-expert users. That is why we propose a recommendation method that takes into account the context of the user. This is done by asking him what are the types of results he would like to obtain with the final model, and notably which performance indicators are most important to him.

With this solution, we want to allow the user to participate in the elaboration of his machine learning model. We aim to avoid the total automation of the machine learning process, which would lead the user to feel left out. Thus, we want to put the user back into the decision process without asking him to rely on too much machine learning knowledge.

In this chapter, we first describe our proposition for taking into account the preferences of the user during the recommendation in Section 5.2. We then describe the dissimilarity

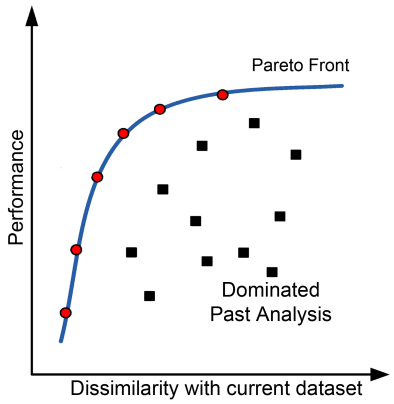


Figure 5.1: Pareto front of the best past analysis according to our two criteria.

measure on which our recommendation system will be based in Section 5.3. This dissimilarity measure is used to test our context-based recommendation system in Section 5.4.

5.2 The recommender system

To make recommendations more accurate, we decide to extend the classical recommendation systems by filtering the performance of a workflow according to the current user's needs. For instance in a prediction task, if a current user has a very high cost of false negatives (like the early diagnosis of a dangerous disease), then we should consider relevant workflows that exhibited good *recall* on similar datasets. Our approach is then to consider criteria able to characterize different aspects of workflow performance to model user preferences. Even considering only problems of supervised classification, many different criteria have been proposed to characterize different aspects of performance, like Cohen's *Kappa* [Coh68], measuring agreement while accounting for the chance of random good guesses, or the more complex *Information Score* from [KB91], measuring the amount of *non-trivial* information produced by the model.

Then, the preference model of a user is represented as a set of performance criteria he is interested in (each of them associated with a weight qualifying its relative importance). For instance, a user who wants to avoid false negatives has *recall* measure as its most important criterion. But it does not mean that *precision* has to be ignored. A higher weight associated with *recall* represents the user preference.

Considering a current user in possession of a dataset and having defined his preferences, the system recommends workflows from the past analysis. This implies access to a base of

past data analysis experiments, where users upload the analysis they perform. One such past analysis then consists of a dataset, upon which was applied a workflow, yielding a result.

The suggested workflow for the current analysis should then be determined according to two criteria:

1. The past analysis must have been produced on a dataset *similar* to the dataset of the current user.

2. Its results, evaluated according to the indicators relevant to the current user, should be satisfactory to the current user.

We thus face a problem of multi-criteria optimization, where both similarity of the dataset and past performance matter. To solve it, we can propose two different approaches:

1. **AverageRecommendation**: Aggregate the two criteria in one. We can normalize those two criteria on the same scale, and use their average as the only criterion. This approach, formalized in algorithm 1, yields a single recommendation that is the *most likely* to suit the user without requiring any additional computation. It however loses most of the information contained in our two original criteria.

2. **ParetoRecommendation**: Consider the full Pareto front of past analysis as a set of recommendations. We can then consider our best possible candidates along with our two criteria (as shown in Figure 5.1), which increases the chances of finding one that suits the user well but requires an additional step to discriminate between candidates. Indeed, supplying the full set of recommendation would probably be useful to expert users, but is most likely to overwhelm a non-expert. This approach is formalized in algorithm 2.

In both those cases, we are effectively creating a *model-based* filtering recommender system, which includes the context of our user into its recommendation during the recommendation process, as described in section 3.1.

In order to develop our algorithms, we introduce a few basics elements and formulas :

First, we need a dissimilarity function akin to the one proposed in [Ray18].Given two datasets D_1 and D_2, a normalized dissimilarity function d will output a positive number, with 0 meaning that the two datasets are completely similar in the sense of the dissimilarity function.

Then, given the results R produced by a workflow W on a dataset D, we need compare those results to the preferences $\mathcal{P}$ indicated by the user through a new function. The

quality of different experiment results according to the current user's preference can then be compared by normalizing the involved criteria and applying the weights representing user preference. With $\mathcal{P}$ The current user's preferences and R an experiment result we will note this quality of the result according to the preferences $\mathcal{Q}_{\mathcal{P}}(R)$.

Algorithm 1 Recommendation by criteria average.

Require: $\mathcal{B} = (D_i, W_i, R_i)_{i \in [1..n]}$ The base of past analysis, each consisting in a workflow W_i applied on a dataset D_i yielding a result R_i

$\mathcal{D}$ The current user's dataset

$\mathcal{P}$ The current user's preferences

for all analysis $(D_i, W_i, R_i) \in \mathcal{B}$ **do**

Compute and normalize $d(\mathcal{D}, D_i)$

Compute and normalize $\mathcal{Q}_{\mathcal{P}}(R_i)$

end for

Find i maximizing $d(\mathcal{D}, D_i) + \mathcal{Q}_{\mathcal{P}}(R_i)$

return W_i

Algorithm 2 Recommendation from Pareto front.

Require: $\mathcal{B} = (D_i, W_i, R_i)_{i \in [1..n]}$ The base of past analysis

$\mathcal{D}$ The current user's dataset

$\mathcal{P}$ The current user's preferences

for all analysis $(D_i, W_i, R_i) \in \mathcal{B}$ **do**

Compute $d(\mathcal{D}, D_i)$

Compute $\mathcal{Q}_{\mathcal{P}}(R_i)$

end for

Compute the Pareto front $F \in \mathcal{B}$ of non-dominated analysis (analyses where neither $d(\mathcal{D}, D_i)$ nor $\mathcal{Q}_{\mathcal{P}}(R_i)$ can be improved in value without degrading the other)

for all non-dominated analysis $(D_j, W_j, R_j) \in F$ **do**

Realize experiment $(\mathcal{D}, W_j, \mathcal{R}_j)$

Compute $\mathcal{Q}_{\mathcal{P}}(\mathcal{R}_j)$

end for

Find j maximizing $\mathcal{Q}_{\mathcal{P}}(\mathcal{R}_j)$ **return** W_j

5.3 Recommending workflow through dissimilarity

In order to test our new recommendation system, we have to use a basis from the state of the art. The most common way to recommend a workflow seems to be the autoML systems, as the one from [Ray18].

The measure of dissimilarity is based on the characteristics of the datasets to be compared. This dissimilarity is computed through two levels of meta-attributes: the difference between each dataset meta-attribute and the difference between each attribute meta-attribute. Those two differences constitute two different dissimilarities, that are then combined to create a more precise dissimilarity measure.

5.3.1 Dataset meta-attributes

To dispose of a large selection of meta-attributes from diverse categories, we use the OpenML platform [Van13]. This platform contains more than a hundred meta-attributes, from different statistical, information-theoretic, and landmarking approaches (complete list available on `http://www.openml.org/`).

5.3.2 Attribute meta-attributes

Individual attributes of datasets can be characterized along with a set of measures, mostly consisting of non-aggregated versions of the previously described *Dataset meta-attributes*. To build our set of *attribute meta-attributes*, we use the 72 measures proposed in [RSV16], able to characterize individuals' attributes. The key idea is to compare attributes of different datasets along their *attributes meta-attributes*. However, as the intuition is to make use of all available information, attributes are compared by most similar pairs: For two datasets A and B, each attribute of A is paired with an attribute of B such as the total dissimilarity of each pair is as low as possible.

5.4 Experiments

5.4.1 Evaluation of the recommendation system

This section describes the experiments conducted to assess our recommender system. The main objective is to show how extending a recommendation system by adding user context can provide workflows that are very similar to what an expert would have done. Our experiments rely on a cross-validation scheme ([Sto74]) over a fixed number of past experiments, iteratively dividing this base in training and testing sets. Our recommender system then

uses the experiments from the training sets to recommend workflows for the experiments in the test set. Those can then be compared to the workflows designed by the users in these experiments.

To perform this comparison, we define an empirical dissimilarity between workflows according to the following intuition: *Two workflows are empirically similar if they exhibit similar performance on similar datasets.* Lets us say we wish to evaluate the dissimilarity between workflows W_A and W_B. We have two sets of experiments $\mathcal{B}_A = (D_i, W_A, R_i)_{i\in[1..n]}$ and $\mathcal{B}_B = (D_j, W_B, R_j)_{j\in[1..m]}$, respectively using W_A and W_B. Considering a dissimilarity between datasets d and a dissimilarity between results d', we can build the sequences $d(D_i, D_j)$ and $d'(R_i, R_j)$. The *correlation* between those sequences then expresses our intuition of empirical workflow dissimilarity: similar workflows tend to perform more the same on similar datasets than on very different ones. Without any hypothesis on the distribution of those dissimilarities, we can compute this correlation using Spearman's rank correlation. We can then define our dissimilarity such as to be null on very similar workflows (Spearman's ρ of 1) and to diverge to infinity on workflow behaving in exact opposition (Spearman's ρ of -1). This dissimilarity between workflows could then be expressed as:

$$\Delta(W_A, W_B) = log_2 \frac{2}{1 + \rho(\ d(D_i, D_j),\ d'(R_i, R_j)\ )}$$

5.4.2 Base of past experiments

In order to be as close as possible to what a real user does, the past experiments are extracted from the OpenML ([Van13]) platform, which makes available millions of machine learning experiments. However, we must consider that it contains many workflows that were generated automatically. Indeed, some users (like scientists developing new machine learning algorithms) generate large benchmarks of experiments, constituting a good portion of the OpenML base. As the main concept we aim to extract is *user expertise*, we should avoid using such experiments, and instead focus on "hand-crafted" ones. As this knowledge is not explicit in OpenML, we decide to consider an experiment "man-made" if the user submitting it submitted at most five experiments involving this particular dataset. Other necessary conditions on the experiments are as follow:

- The dataset must be available to allow computing of dataset dissimilarities.
- The considered evaluation criteria must be available to allow adequation to user preferences.

- OpenML must contain at least 50 experiments using the same workflow to allow computing workflow dissimilarities.

- Each experiment should consider a different dataset to represent the fact that a user coming with a new dataset does not already find experiments using it in the base.

We found a total of 60 experiments on OpenML satisfying to all those conditions (see *Resources* section for listings). They then constitute our base of past experiments.

This base gives us the possibility to evaluate our recommender system on a reasonably diverse (albeit still small) base of past experiments. Expanding it would be of great interest, but the cost of producing acceptable experiments is prohibitive, while the whole OpenML base (more than 8 million experiments) only yielded few candidates. To investigate the performance of the recommender system on bases with different profiles, we thus define subsets of our base advertising particular characteristics.

- First we investigate how the diversity of datasets affects recommendation performance. Two subsets of the base are defined by choosing one "central" experiment and isolating the subset it forms with its closest neighbors (in the sense of dataset dissimilarity) from the subset of experiments with much more dissimilar datasets. This "central" experiment is then chosen to maximize the difference in average dataset dissimilarity between the two subsets. The resulting partition results in a subset containing quite similar datasets, while the others are very different. The average internal dissimilarities (average dissimilarity between all datasets of the subset's experiments) of those subsets can be found in table 5.1. It can be viewed as a measure of dataset diversity in the subset.

- Second, we can investigate how the diversity of workflows affects recommendation performance. The process is very much the same, but substituting our dissimilarity between workflows to the one on datasets. This then results in a subset of experiments with very similar workflows and another with more different ones. The average internal workflow dissimilarities of those subsets can also be found in table 5.1.

- Third, this partition can also be made by dissimilarity of results. Here, a normalized Manhattan distance on the available evaluation criteria enables splitting our base into two subsets, the first with similar profiles of results and the second having them more different. Their average internal distance between results can also be found in table 5.1.

Table 5.1: Average dissimilarities on the subsets.

Partitioning on	Dataset	Workflow	Results
Closest set	0.122	0.429	0.242
Full base	0.153	0.841	0.266
Farthest set	0.170	1.252	0.300

5.4.3 Baseline

We compare our system against a baseline of two simple recommendation processes:

1. **Random**: Randomly recommends a workflow from one of the past experiments. The purpose of this baseline is to roughly evaluate the performance achievable without using any of the available information. Due to the obvious randomness of this approach, its results were averaged over ten repeats of the whole evaluation scheme.

2. **BestPerformance**: Search the base of past experiments for the one with the best result according to the *current* user's preferences, and recommend the associated workflow. For instance, a user requiring only high accuracy is given the workflow that scored the highest accuracy in the base. The purpose of this baseline is to evaluate the performance achievable using only performance-related information.

5.4.4 Experimental setup

To evaluate the performance of our recommender system, we perform leave-one-out cross-validation on this base. We successively consider each experiment of the base as a new user in need of assistance and use the rest of the base to recommend a workflow. The dissimilarity between the recommended workflow and the workflow originally used in the experiment measures the performance of the recommendation.

A threshold of this dissimilarity must be set to characterize appropriate recommendations and compute a precision score. For a given threshold, a recommendation is successful (positive) if the dissimilarity between the recommended workflow and the workflow originally used in the experiment is lower than that threshold, and failed (negative) otherwise. Thus we can compute a standard precision score (number of positives over the number of recommendations) for each value of the threshold. We plot the achieved precision depending on the set threshold, much in the same way as a *ROC* curve. Note that a recall score is not necessary, given that our system is always able to recommend a workflow.

5.4.5 Discussions

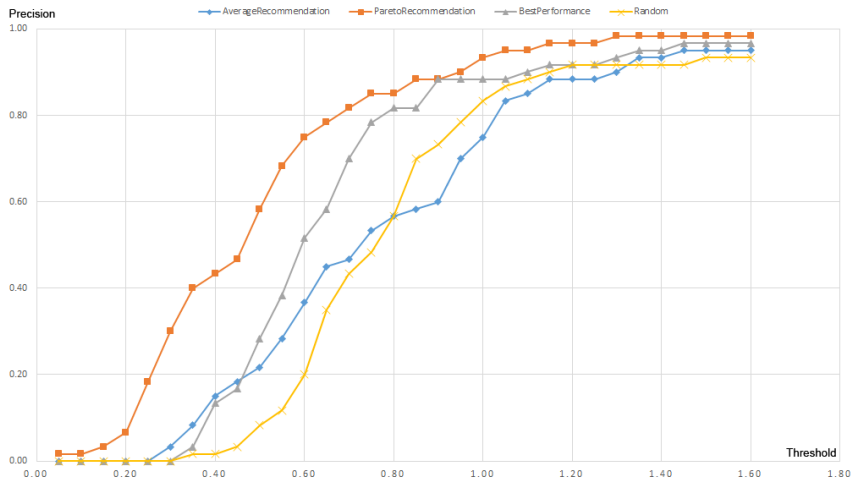


Figure 5.2: Achieved precision by threshold, on the full base.

The results are presented Figures 5.2 and 5.3. These results give the precision scores for thresholds of dissimilarity between 0.0 and 1.6. Indeed, thresholds greater than 1 already consider most workflows to be similar, and pushing the threshold further only results in all methods having a precision of 1 (any recommendation being *similar enough* to the original workflow).

Figure 5.2 describes the results obtained while performing cross-validation on the full base. We can note that all the curves converge since, for increasingly higher thresholds, more and more workflows are considered similar. The orange curve, depicting the recommender system investigating the whole Pareto front, gives the best results for any threshold value, even in the case of a low threshold. For instance, considering a threshold of 0.6, the precision score is about 0.75, whereas all the other approaches are under 0.52. In particular, the Pareto recommendation continuously overperforms the recommendation based on the best available performance. This result indicates that the use of the contextual dataset to filter the relevant experiments is effective. However, when analyzing the blue curve, representing our approach based on an average of the criteria (instead of the Pareto front), the performance appears much lower. This result advocates precaution when using contextual information about the dataset: it brings more accurate recommendations but can be detrimental if factored in without further considerations.

Figure 5.3 refines the previous results by partitioning the base of past experiments in subsets, according to the diversity of the dataset, workflows, and results in the base (as seen in 5.4.2).

When segregating by dataset diversity, we can see that the *ParetoRecommendation* still

outperforms the others, whether the datasets in the base are very similar or very different. When comparing to its results on figure 5.2, the *ParetoRecommendation* does not appear to be very sensitive to dataset diversity. This result indicates that our system can recommend relevant workflows in a context where past datasets would not be similar to the current one.

When segregating by workflow diversity, we notice that the results on the set with close workflows reach the maximum very fast. This result is expected since the probability of drawing a similar workflow is greater when most are similar to each other (that is why the random approach outperforms the other approaches with lower thresholds than for other cases). Considering the farthest set, the *ParetoRecommendation* outperforms again the other methods, but with a generally lower precision score (with a threshold of 0.6, the precision score is about 0.59, while we reached 0.75 on the full base). Again, this can be expected, as it is harder to provide similar workflows when most are very different, but observing such a small loss in performance (compared to the other approaches) is encouraging.

When segregating by result diversity, we can identify that precision scores using the closest set seem worse than using the farthest set. For instance, for a threshold of 0.6, the *ParetoRecommendation* scores 0.64 using the closest set, and 0.82 using the farthest set. Indeed, results diversity appears to be an important property of the base, as it increases the chance of finding past experiments suiting the user's preferences. We should though note that similar results do not necessarily mean similar workflows were used. For example, we can maximize a recall score using different data-mining algorithms. Finally, we can notice that the *ParetoRecommendation* still appears to outperform the other methods on most of the space, ensuring relative robustness to most kinds of skewed bases. This bodes well toward its result on potentially much larger bases, even when operating conditions restrict its diversity (for instance, a base built on the past experiments of particular researchers can have low dataset diversity, as most are highly similar).

5.5 Conclusion

In this chapter, we extended a recommendation system for recommending workflows by adding user preferences. This system is based on information obtained from the contexts of the dataset to analyze and the expected results of the analysis. The dataset context is used through a dissimilarity measure based on important properties characterizing a dataset and its topology. The expected result context is used through a set of user preferences allowing

the characterization of the results along with any number of evaluation criteria.

Two versions of the recommender system are proposed. Indeed, to consider the two contexts, we faced a problem with multi-criteria optimization. Thus, the first version considers an average of the two criteria (dataset likeness and result quality) to elicit recommendations. The second instead considers the whole Pareto front along with those criteria.

The two versions of the recommender system are tested on real workflows (coming from the OpenML platform) and compared with two simple recommendation processes (random generation and best past performance). The experiments show that our recommender system is based on the Pareto front outperforms, in terms of precision, all the other approaches, even in the case of low threshold (where the dissimilarity is the most demanding). Our approach is also efficient in managing past datasets which are not so similar to a current dataset, or other bases offering skewed distributions.

Our short-term perspective is to better take into account the user profile, especially his expertise level to form a new axis of context, in order to improve user satisfaction concerning the recommendations.

Our long-term perspective is to propose a complete solution aimed at domain experts integrating our recommender system to automatically propose workflows that could match with the user needs. It also supposes to be able to graphically represent this type of recommendations to the user.

References

[Coh68] Jacob Cohen. “Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit.” In: *Psychological bulletin* 70.4 (1968). Publisher: American Psychological Association, p. 213.

[KB91] Igor Kononenko and Ivan Bratko. “Information-Based Evaluation Criterion for Classifier’s Performance”. In: *Machine Learning* 6.1 (Jan. 1991), pp. 67–80.

[Ray18] William Raynaut. “Perspectives de Méta-Analyse pour un Environnement d’aide à la Simulation et Prédiction”. francais. Thèse de doctorat. Toulouse, France: Université de Toulouse, Université Toulouse III-Paul Sabatier, Jan. 2018.

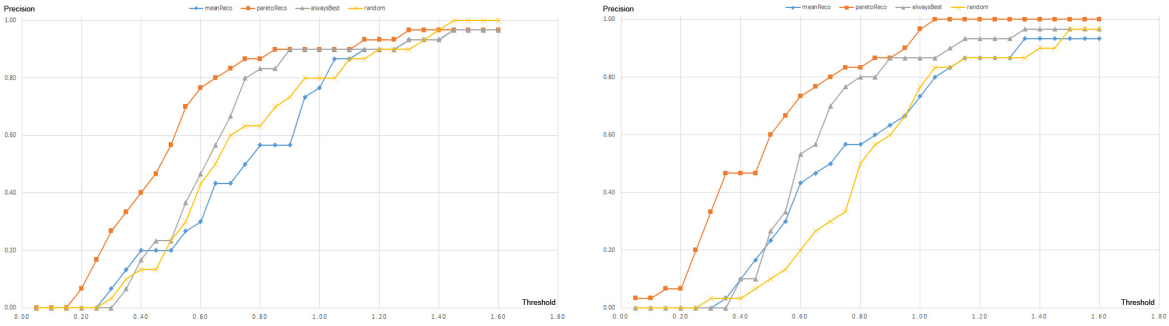
[RSV16] William Raynaut, Chantal Soule-Dupuy, and Nathalie Valles-Parlangeau. "Meta-Mining Evaluation Framework : A large scale proof of concept on Meta-Learning". In: *29th Australasian Joint Conference on Artificial Intelligence*. (Classée B par core.edu.au). Springer, Dec. 5, 2016, pp. 215–228.

[Sto74] M. Stone. "Cross-Validatory Choice and Assessment of Statistical Predictions". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 36.2 (1974), pp. 111–147. DOI: 10.2307/2984809.

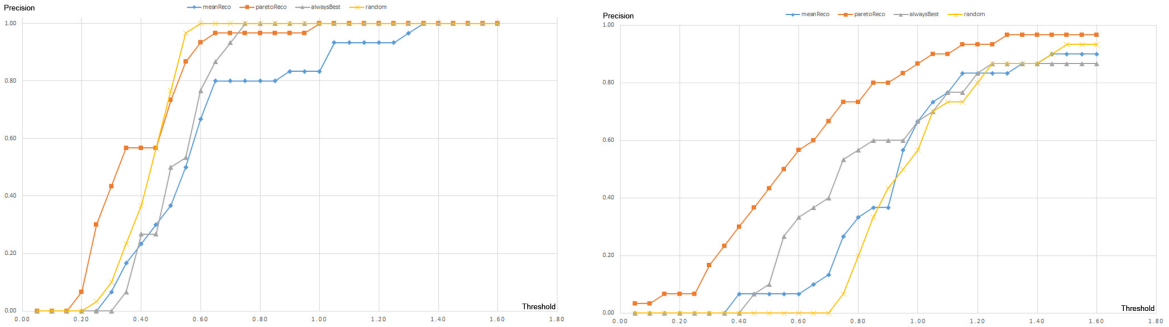
[Van13] Joaquin Vanschoren et al. "OpenML: Networked Science in Machine Learning". In: *SIGKDD Explorations* 15.2 (2013). Place: New York, NY, USA Publisher: ACM, pp. 49–60. DOI: 10.1145/2641190.2641198.

Closest set Farthest set

Partitioning on Dataset



Partitioning on Workflow



Partitioning on Results

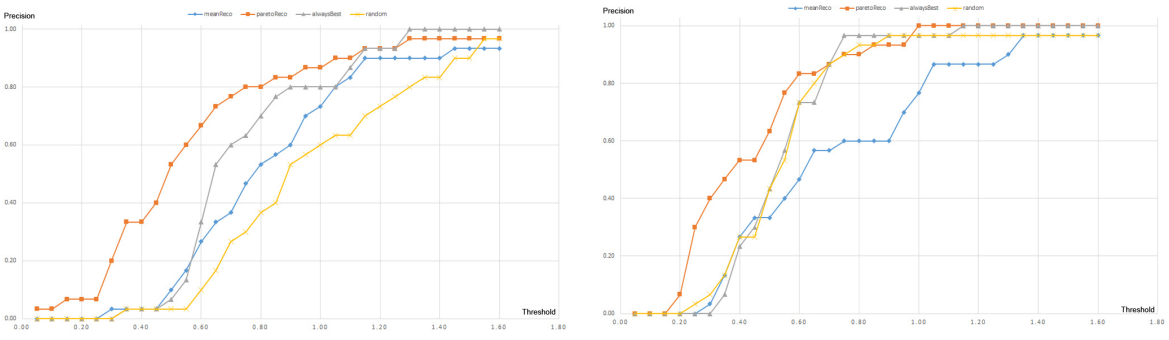


Figure 5.3: Achieved precision by threshold, on the different subsets.

A new, a vast, and a powerful language are developed for the future use of analysis, in which to wield its truths so that these may become of more speedy and accurate practical application for the purposes of mankind than the means hitherto in our possession have rendered possible.

Ada Lovelace

Analysis gave me great freedom of emotions and fantastic confidence. I felt I had served my time as a puppet.

Hedy Lamarr

Chapter 6

Explaining single predictions through coalitions

6.1 Introduction

As we have seen in Part I, machine learning tasks are difficult to perform for a domain expert user, someone having a deep knowledge of the data to analyze, but without a background in data analysis. This is mainly caused by the "black box" feeling that a trained predictive model can leave on its user, undermining its use.

This means that even if a model has been trained by an expert, its prediction can suffer from a lack of confidence from the end-user. This user may not be inclined to believe in a prediction obtained in a way he does not understand. This trust issue has been studied, among other works, in [RSG16].

Thus, automatically recommending a machine learning workflow is not enough to assist a non-expert user. Instead, to **help a non-expert understand and analyze the results produced by a data analysis process**, we need a system which allows the user to understand what is being recommended, and what he is using.

Many explanation methods exist in the literature (as we have seen in Chapter 4) to overcome the problem of accessibility to prediction results. These methods are mainly devoted to the global model explanation, which explains how a predictive model functions as a whole. These methods are not relevant when a domain expert user has to study the behavior of particular dataset instances over a predictive model. For example, a physician wanting to perform a cohort study of his patients might need an explanation of prediction for every single patient, rather than a global one.

To fulfill that need, some researchers have studied the possibility of explaining single

instance prediction of a model, e.g. [SK10] and [CMB18]. However, these methods may be specifically designed for a single model type (as in [LL17b]), which limits their use. Methods designed to be applicable for all types of models will instead lack precision (as in [ŠK08a]), or overcoming this lack of precision will make their algorithms very long to apply (as the one proposed in [SK10]). This means the field of single instance prediction explanation needs to be developed further, to make it usable by everyone. In this chapter, we propose a new method aiming to strike a balance between precision and computation time, in order to make machine learning more understandable for everyone.

In the first part of this chapter, Section 6.2 we compare two basic methods developed in [ŠK08a] and [SK10]. Those methods respectively estimate the influence of a dataset attribute on a model prediction by assessing the information brought by the attribute in the model. This leads us to find a basis for a prediction explanation system. From this basis, we then propose our own prediction explanation system based on Shapley values in Section 6.3. As it takes into account the influence of all the combinations of attributes of a dataset, our method achieves a very satisfying precision. The problem is that this new explanation method pays its precision at the cost of a long computation time. That's why we decide to identify the groups of attributes that bring the most information to the explanation and calculate only those groups. Those simplification methods are described and tested in Section 6.4. Those first solutions lead us to research more possible important group identification methods. Those are described and tested in Section 6.5. All of the methods we propose are finally evaluated and compared in Section 6.6.

6.2 Deciding for a basis

As stated in Chapter 4, *additive* explanation methods produce a vector of weights, with each weight linked to an attribute of the dataset. The weight of an attribute represents its importance for the prediction of the model and provides a good description of how the model "thinks" to the user. For those reasons, we decide to use additive explanations as described in [LL17a] for our system. One big limit to the use of a single-instance additive prediction explanation is their computation time. Thus, in order to make efficient use of it, we have to optimize its computation.

To start developing a faster *additive* explanation method, we start by selecting an algorithm from the literature and then aim to reduce its complexity without losing too much information. For this, we compare two methods developed by Strumbelj et al. in [ŠK08a] and [SK10], as they are similar in their design (they both relies on the comparison

between two predictions of the same instance but with a modification on the studied attribute), but different in their interpretation. This difference relies on their methods of calculation: the first one relies on what is lost when an attribute is removed, while the other one looks at what is gained when an attribute is added.

6.2.1 Additive method versus substractive method

Definition 7 *Given a dataset D of instances and a set of n attributes $A = \{a_1, .., a_n\}$, each attribute being either numeric or nominal, its possible values can then be expressed as integers for nominal attributes or real number for numeric attributes. Each instance $x \in D$ is defined by the values of each of its attributes : $x = \{x_1, ..., x_n\}, \forall i \in 1..n, x_i \in \mathbb{N} \vee x_i \in \mathbb{R}$. We want to explain a predictive model, based on the function $f : D \to [0, 1]$, whose result is the confidence score in the classification of the instance x for a class C, as predicted by the model.*

- **substractive method**

One of the first definitions for classification explanation is proposed in [ŠK08a]. According to their method, the influence of an attribute a_i on the classification of a given instance is defined as the difference between the classifier prediction (with a_i) and its prediction without the knowledge of attribute a_i. Thus, given a dataset of instances described along the attributes of A, the influence of the attribute a_i on the classification of an instance x by the classifier confidence function f on the class C can be represented as:

$$inf^C_{f,a_i}(x) = f(x) - f(x \backslash a_i) \tag{6.1}$$

Where $f(x \backslash a_i)$ represents the probability distributions for a classification of the instance x by the classifier f without knowledge of the attribute a_i. We name this method as the *information loss method* (shortened as *loss method*).

- **Additive method**

In more recent works, as in [SK10], another possible formula is based on the information brought by an attribute in the dataset:

$$inf^C_{f,a_i}(x) = f(x_{a_i}) - f(\varnothing) \tag{6.2}$$

Where $f(x_{a_i})$ represents the probability that the instance x is included in the class C with only the knowledge of the attribute a_i (according to the predictive model) and $f(\varnothing)$

is the probability that instance is in the class C with no knowledge of the instance (thus it is the proportion of instances of this class among the whole dataset). We name this method as the *information gain method* (shortened as *gain method*). In order to simulate the absence of an attribute, the authors of [ŠK08a] theorize possible approaches, among which we selected to retrain the classifier without the corresponding attribute.

- **Comparing the two methods**

As an illustration, and to ease interpretations, we apply these two methods in a simple ID3 decision tree [Qui86], trained on the well-known Fisher's $Iris$ dataset[1]. As a decision tree is a naturally interpretable model, and the Iris dataset is well studied in the literature, it is easy to compare and interpret the two methods and detect eventual problems. We use Weka [Hal09] and OpenML [Van13] to perform the data management and model training while ensuring the reproducibility of all experiments. To estimate the reliability of *loss* and *gain* methods, we apply 5-fold cross-validation on the $Iris$ dataset. The explanations of both methods are generated on the validation set, for each iteration of the cross-validation. We generate thus prediction explanations of Weka's $J48$ tree classifier, for the whole $Iris$ dataset. Then, we compare those explanations to the decision tree and get a general sense of the explanation's accuracy.

Each instance of the Iris dataset is composed of four attributes: *petal length*, *petal width*, *sepal length* and *sepal width*. Each instance is included in one of these three classes: $Iris$ $Setosa$, $Versicolor$, or $Virginica$.

By a simple look at the trained decision tree Figure 6.2, we see the influence should be zero for the sepal length and width attributes, as they are not used by the tree at all. Moreover, the $Setosa$ class instances should only be influenced by petal width, as it is the only attribute used to classify them. For *Iris Virginica* and *Versicolor*, we can expect a high influence from petal width, and a lower from petal length influence, yet still significant as these two attributes are used. We can now compare these expectations to the results indicated in Table 6.3. We see the *loss method* does not behave as expected for the $Setosa$ instances, as all the attributes are being given an influence equals to 0. This can be understood by looking at the representation of the concept learned by the tree Figure 6.1. We note that, by removing one of the attributes between petal length and petal width, it remains possible to separate the $Setosa$ class linearly from the others using only the petal length, and still maintain a 100% confidence in the classification. Thus, each attribute is considered as inconsequential by the *loss method*, when classifying $Setosa$

[1]Iris, Fisher: https://en.wikipedia.org/wiki/Iris_flower_data_set

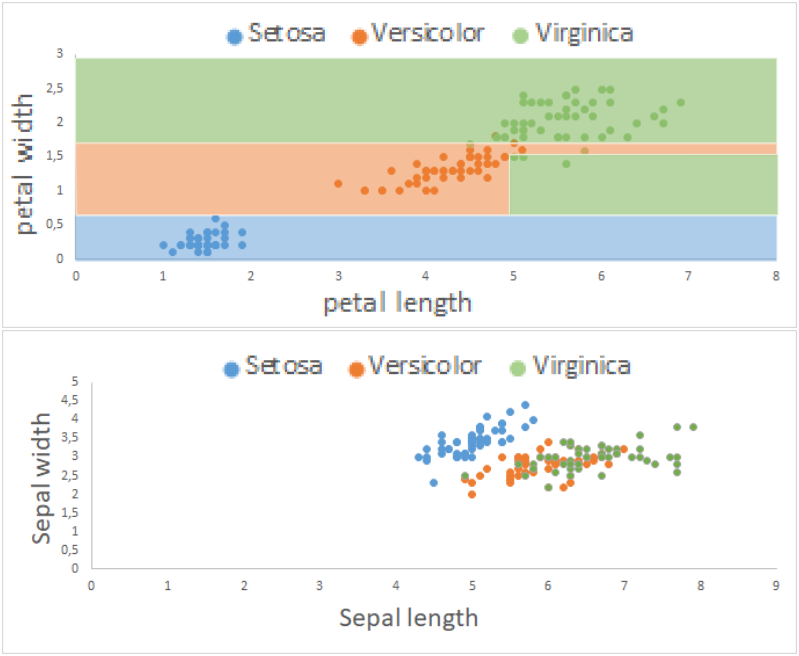


Figure 6.1: Repartition of the 3 different classes by petal length and width, with their corresponding generalization according to the decision tree.

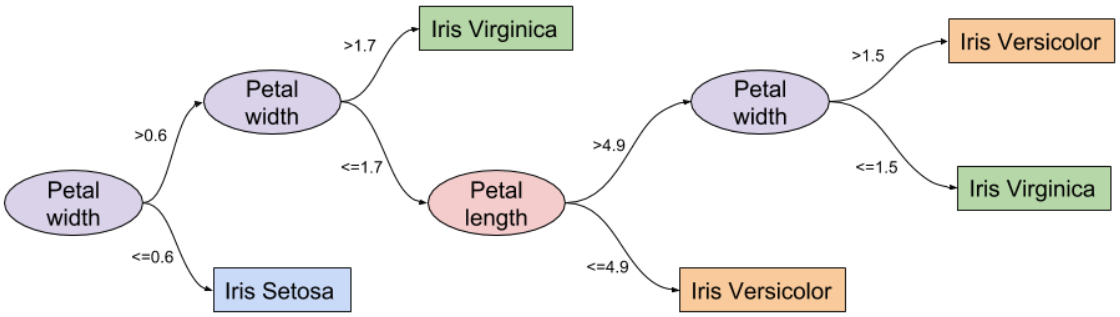


Figure 6.2: A decision tree trained on Iris.

	Loss method				Gain method			
	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
Setosa	0	0	0	0	0.211	0.104	0.316	0.316
Versicolor	0	0	0.835	0.135	0.037	0.077	0.308	0.373
Virginica	0	0	0.047	0.691	0.150	0.019	0.355	0.375
Average	0	0	0.305	0.284	0.131	0.066	0.326	0.356

Figure 6.3: Average influence of the different attributes according to each explanation method, for each class of instances.

instances. This implies that for every dataset in which two attributes carry very similar information, the *loss method* is unable to generate a satisfying and accurate explanation. The *gain method*, on the other hand, gives importance to all the four attributes about *Setosa* instances. Minimal importance is given for sepal length and width, unlike the petal length and width. It is easily understandable by observing the graphs of the repartition of the different classes (Figure 6.1). We note the petal length and width can easily separate the three classes, but the sepal attributes are less defined in their separation. This is especially true for the $Versicolor$ class, as it is mixed with its two adjacent classes. Thus, the *gain method* seems to be closer to what the decision tree is doing when being trained. Finally, we conclude the main difference between the two methods relies on the fact they are not trying to calculate the same thing: the *loss method* is based on the information lost by the model when removing an attribute, while the *gain method* is based on the information brought by each attribute. Yet, we remark the *loss method* has an aberrant behavior when confronted with two attributes bringing the same information. Thus, the *gain method* seems the best proposal for a prediction explanation. But this method has its flaws: it only takes into account the information brought by each attribute, independently. In a dataset, attributes may be interdependent. This is more likely when said datasets have been created by someone who is not an expert in data analysis, as they are prone to ignore that redundant information can prove to be harmful to diverse data analysis tasks. Thus, our next objective is to consider the influence of a group of attributes on the prediction of the model.

6.3 Finding a new solution

6.3.1 Complete method

To answer the problems of interaction between attributes, we propose to take inspiration from the work of [SK10]. We are here in a framework close to the situation of a game called "coalitions", where each group of attributes can influence the prediction of the model. Therefore, we cannot consider each attribute as independent, but all the possible combinations of attributes. The influence of an attribute is measured according to its importance in each coalition. We can then refer to the coalition games as defined by Shapley in [Sha53]: A coalitional game of N players is defined as a function mapping subsets of players to gains $g : 2^N \mapsto \mathbb{R}$. The parallel can easily be drawn with our situation, where we wish to assess the influence of a given attribute *in every possible coalition of*

attributes. We then look at not only the influence of the attribute but also its use in all subsets of attributes. We thus define the *complete influence* of an attribute $a_i \in A$ on the classification of an instance x : given a dataset of instances described along the attributes of A, the *complete* influence of the attribute a_i on the classification of an instance x by the classifier confidence function f on the class C is dependant on the influence of all the possibles subgroups $A' \subseteq A$ which does not contain a_i. Thus, the *complete* influence of a_i is :

$$\mathcal{I}^C_{a_i}(x) = \sum_{A' \subseteq A \setminus a_i} p(A', A) * (inf^C_{f,(A' \cup a_i)}(x) - inf^C_{f,A'}(x)) \tag{6.3}$$

With $p(A', A)$ a penalty function accounting for the size of the subset A'. Indeed, if an attribute changes a lot the result of a classifier, in a large group of attributes, it can be considered as very influential compared to the others. On the opposite, an attribute changing the result of a classifier, whereas this classifier is based on a few attributes, cannot be considered to have a decisive influence. The Shapley value [Sha53] is a promising candidate and defines this penalty as:

$$p(A', A) = \frac{|A'|! * (|A| - |A'| - 1)!}{|A|!} \tag{6.4}$$

This *complete influence* of an attribute now takes into consideration its importance among all the possible attribute configurations, which is closer to the original intuition behind attributes' influence. However, computing the *complete influence* of a single instance is extremely computationally expensive, with complexity in $\bigcirc(2^n * l(n, x))$, with n the number of attributes, x the number of instances in the dataset, and $l(n, x)$ the complexity of training the model to be explained. It is then not practical to use the *complete influence*. Consequently, it becomes necessary to seek a more efficient way to explain predictions. Although the *complete influence* is too computationally heavy, it can be considered as an excellent baseline ([SK10]). Thus, we can evaluate other explanation methods by studying their differences with the *complete influence*.

Example 12 *As depicted in Figure 6.4, the influence of an attribute depends on its influence alone, but also each possible groups of attributes containing it. As in the Figure, for a dataset with 4 attributes A B C D, the influence of the attribute A is composed of the influence of* $\{A\}$ *alone, along with the influences of the groups* $\{A, B\}$,$\{A, C\}$,$\{A, D\}$,$\{A, B, C\}$, $\{A, B, D\}$,$\{A, C, D\}$ *and* $\{A, B, C, D\}$.

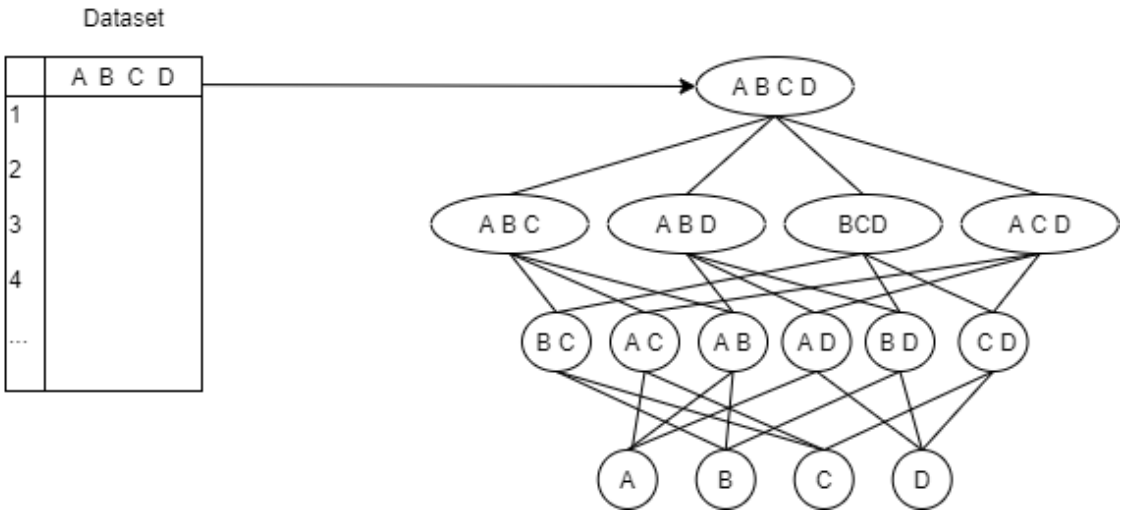


Figure 6.4: Depiction of the groups calculated by the complete method for a 4 attributes dataset. Each possible combination of attributes is calculated to ensure an influence value as close to the reality as possible.

6.4 Approximating the complete method

The complete method produces a far more accurate explanation of the model's way of functioning, but the number of subgroups to calculate grows exponentially with the number of attributes of the dataset. Thus, it takes a very long time to calculate the complete influence for a dataset with too many attributes. To circumvent this limitation, we have to find a good compromise between the precision of the explanation and its computation time. Those compromises will consist of finding the most pertinent groups of attributes to calculate, and determining which groups can be ignored.

6.4.1 K-complete method

An approximation of the *complete influence* has to remain accurate and practical, as much as possible. For this we cannot fully rely on recent works (e.g. [SK10] and [LL17a]), as explained in Section 3. In particular, looking for a subset of all the subgroups could be more practical in terms of complexity. This solution should produce explanation, a priori, more accurate than the basic consideration of independent attributes (*linear influence*). We consider then the *depth-k complete influence* defined as the complete influence, but ignoring the groups of attributes A' with a size superior to k :

$$\mathcal{I}_{a_i}^{C_k}(x) = \sum_{A' \subseteq A \setminus a_i,\ |A'|<k} p_k(A', A) * (inf_{f,(A' \cup a_i)}^{C}(x) - inf_{f,A'}^{C}(x)) \tag{6.5}$$

$$p_k(A', A) = \frac{|A'|! * (|A| - |A'| - 1)!}{k * (|A| - 1)!} \tag{6.6}$$

In particular, we can note that the *linear* influence is actually identical to the *depth*-1 *complete influence*. The intuition behind this approach is to eliminate the larger groups, which have a lesser impact on the Shapley value while being the most costly to calculate. We then hope to achieve a better calculation time without losing too much information.

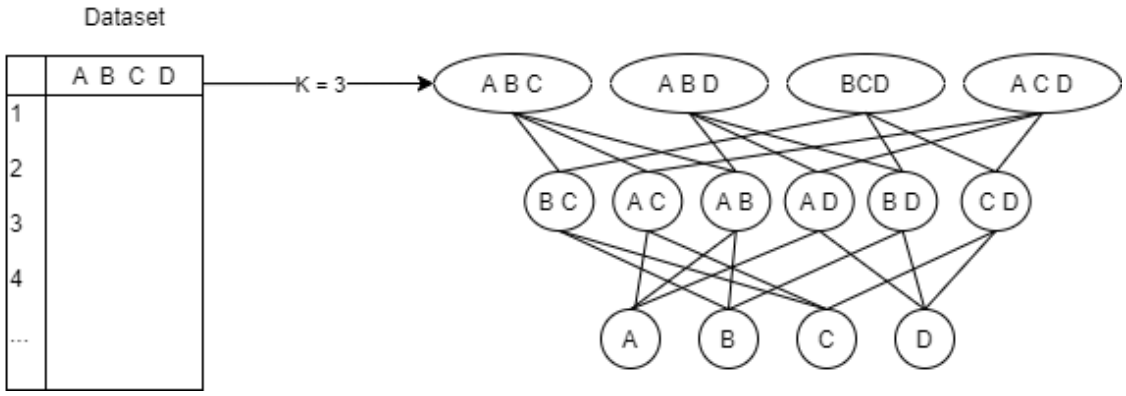


Figure 6.5: Depiction of the groups calculated by the k-complete method for a 4 attributes dataset. The group size is limited by the k parameter : here, the groups maximum size is 3.

Example 13 *As depicted in 6.5, for the same dataset with 4 attributes and a parameter* $k = 3$*, the total influence of the attribute A only depends on the influence of A alone and the groups of attributes containing A and with a maximum size of* 3 *:* $\{A, B\}$,$\{A, C\}$,$\{A, D\}$, $\{A, B, C\}$,$\{A, B, D\}$ *and* $\{A, C, D\}$

6.4.2 Coalitional method

Another possible approach is to identify the attributes having an interaction between them. We can obtain a grouping such as $G = \{\{a_1, a_3\}, \{a_2, a_5, a_8\}, \{a_4\}...\}$. With such groupings of attributes, it becomes possible to consider only the attributes of a subgroup, without having to consider every possible attributes' combination. It is important to note that the groups don't necessarily have to be exclusive, which mean an attribute a_i can be found in multiples groups of G. We then obtain a *coalitionnal influence* of an attribute a_i : Given G_{a_i}, the subset of G containing all the attributes groups $g \in G$ such as $a_i \in g$

$$simple\mathcal{I}^C_{a_i}(x) = \sum_{g' \subseteq g \setminus a_i,\ g \in G_{a_i}} p_c(g', g, G_{a_i}) * (inf^C_{f,(g' \cup a_i)}(x) - inf^C_{f,g'}(x)) \tag{6.7}$$

$$p_c(g', g, G_{a_i}) = \frac{|g'|! * (|g| - |g'| - 1)!}{\sum_{g \in G_{a_i}} |g|!} \tag{6.8}$$

Given the fact that we can set a maximum cardinal c for our subgroups, the complexity is now, in the worst case, $O(2^c * \frac{n}{c} * l(n, x)) \approx O(n * l(n, x))$. This method calculate less groups than the *depth-k complete influence*, but tries to make up for it by only grouping the attributes actually related to each other. In order to determine which attributes seem to be related, we can use an automated corellation detection algorithm, as in [Hen14].

Model-based coalition

Our first proposed implementation of an attribute coalition algorithm is the model-based coalition. The attribute groups are created by using the model itself to detect interacting attributes. In this approach, no correlation is detected, but only interaction in the sense of the model usage of the attributes. This is done by randomizing the values of the dataset and studying the evolution of the model predictions. It consists of measuring the differences of predictions on the whole dataset before and after the randomization. When attributes are considered to be part of the same group, their values are swapped together with the values of another instance, classified by the model as the same class as the starting instance. Each attribute outside of the group has its value swapped completely randomly. Once this has been done, the new instances are classified by the model. The ratio of differences between the old and the new classification is called fidelity. A higher fidelity meaning a lower variation of the predictions. At each iteration, the attribute which removal lowers the less the fidelity is removed until it is not possible to keep the fidelity above a fixed threshold. Then the group is considered as fixed. This attribute grouping algorithm has been developed in [Hen14] and is detailed in Algorithm 3.

Example 14 *Given the same dataset with 4 attributes, we apply the algorithm from [Hen14] on the dataset. It aims to build groups as small as possible such as when*

- *The values of the grouped attributes are randomized* ***inside*** *of their original instance class ;*
- *The values of the non-grouped attributes are randomized completely ;*
- *This randomization is applied to the whole dataset.*

The predictions of the model for the whole dataset do not vary more than a threshold percentage of δ.

Algorithm 3 Model-based coalition extraction.

Require: Sensitivity parameter $\delta > 0$, the number of attributes m, and a fidelity function $fid()$. Two auxiliary functions $L(X) = \bigcup_{i \in X} \{\{i\}\}$ and $F(X) = L(\bigcup_{Y \in X} Y)$, which produces sets of singletons (e.g. $L(\{1,2,3\}) = F(\{\{1,2\},\{3\}\} = \{\{1\},\{2\},\{3\}\})$

Ensure: σ a coalition of attributes

$\sigma \leftarrow \{\}$

$R \leftarrow \{m\}$ ▷ R contains a group to test for

$A \leftarrow \{\}$ ▷ A contains the removed attributes

$\Delta \leftarrow fid(L([m])) + \delta$

while $R \neq \{\} or A \neq \{\}$ **do**

if $A = \{\}$ and $fid(\{R\} \bigcup F(\sigma)) < \Delta$ **then**

▷ if we are already below Δ before removing any attribute assign the remaining attributes to singleton groups

$\sigma \leftarrow \sigma \bigcup L(R)$

$R \leftarrow \{\}$

$A \leftarrow \{\}$

else

▷ Find an attribute j whose removal from R decreases the fidelity least

$j \leftarrow argmax_{j \in R} fid(\{\{R \backslash \{j\}\} \bigcup \{\{j\}\} \bigcup \{A\} \bigcup F(\sigma))$

if $|R| = 1$ or $fid(\{\{R \backslash \{j\}\} \bigcup \{\{j\}\} \bigcup \{A\} \bigcup F(\sigma))$ **then**

▷ If the fidelity drops below Δ add the group of attributes to the results and look for the next group of attributes

$\sigma \leftarrow \sigma \bigcup \{R\}$

$R \leftarrow A$

$A \leftarrow \{\}$

else

▷ If the fidelity stays above Δ continue removing the grouping R

$R \leftarrow R \backslash \{j\}$

$A \leftarrow A \bigcup \{j\}$

end if

end if

end while

return σ

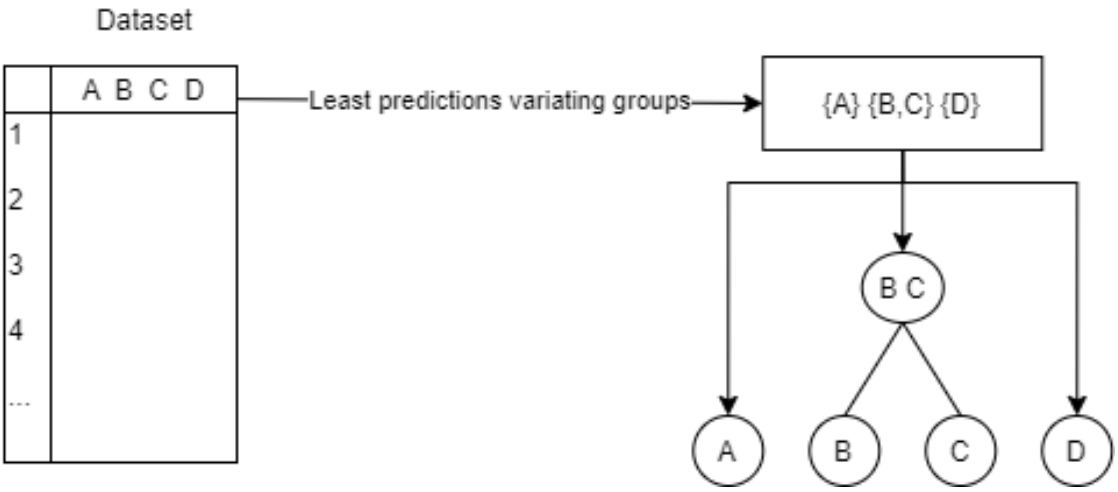


Figure 6.6: Depiction of the groups calculated by the model based coalition method for a 4 attributes dataset.

In the first iteration, the algorithm finds that the smallest group of attributes that makes the predictions varies less than the threshold δ is $\{B, C\}$. Removing C or B makes the predictions vary more than the threshold and as such, the algorithm stores $\{B, C\}$ as a first group.

Now the algorithm tries to build another such group with the remaining non grouped attributes and find that $\{A, D\}$ makes the predictions vary too much. Since the biggest remaining possible group is already making the predictions vary more than the threshold, all the non grouped attributes are considered as singletons, resulting in the grouping $\{\{A\}, \{B, C\}, \{D\}\}$.

This grouping is then used to determine how each attribute has its influence calculated. As an example, the total influence of A only consists of the influence of the singleton $\{A\}$, while the total influence of B is composed of the influences of $\{B\}$ and $\{B, C\}$. Similarly, the total influence of C includes the influences of $\{C\}$ and $\{B, C\}$, and as D is in a singleton, its influence only takes into account the influence of $\{D\}$. Those groups are depicted in Figure 6.6

6.4.3 First experiments

To determine if it is possible to generate a satisfactory approximation of the influence of an attribute with the new *depth-k complete influence* and the *coalitionnal influence*, it is necessary to assess the number of attribute's combinations we need to take into account before being sufficiently near to the *complete influence* defined in 6.3. Moreover, we need to assess if the results of the *depth-k complete influence* produce better explanations than

linear and *coalitional* influences, in view of its higher computation cost. These are the objectives of our next section.

Experimental protocol

Our experiments are run on the OSIRIM[2] cluster. This cluster is equipped with 4 AMD Opteron 6262HE processors with 16 x 1,6 GHz cores, for a total of 64 cores, and 10 x 512 GB of RAM. Our tests are realized from the data available in the Openml platform ([Van13]). We selected the biggest collection of datasets[3] on which classification tasks have been run. We also consider six classification tasks: Naïve Bayes, Nearest Neighbors, J34 Decision Tree, J34 Random Forest, Bagging Naïve Bayes, and Support Vector Machine. Due to the heavy computational cost of the complete influence (considered as the reference of our experiments), we selected the datasets having at most nine attributes.

Thus, a collection of 324 datasets is obtained. Considering the six types of workflows, we have a total of 1944 runs. For each of those runs, we generate each type of influence proposed in this chapter, for each instance of the 324 datasets: the *complete* influence for the baseline, along with the *linear*, *coalitional* and *k-complete* influences. The *k-complete* influences are generated for every possible values of k (from 2 up to the number of attributes of the dataset).

The *coalitional* influences are generated using subgroups of attributes. Here, these subgroups are produced using the algorithm described in [Hen14], which is based on an $\alpha \in]0, 0.5[$ parameter (small values of α resulting in smaller subgroups, and high values in bigger ones). We generate the possible subgroups with 5 different values of α to study the influence of subgroup size.

To compare the different explanation methods, we consider the explanation results as a vector of attribute influences noted $\mathcal{I}(x) = [i_1, ..., i_n]$ with n the number of attributes in the dataset. Thus, each of the attributes a_k is given an influence $i_k \in [0, 1]$ by the method $\mathcal{I} : \forall k \in [1..n], i_k = \mathcal{I}_{a_i}(x)$. We then define a difference between two vectors of influences i, j as the normalised euclidian distance:

$$d(i, j) = \frac{1}{2\sqrt{n}} \sum_{k=1}^{n} \sqrt{(i_k - j_k)^2} \tag{6.9}$$

Considering this formula, we define an error score based on the difference between an explanation method and the *complete* influence method. Given an instance x, an explanation

[2] http://osirim.irit.fr/site/en

[3] Available in `https://www.openml.org/s/107/tasks`

method $\mathcal{I}(x)$, and the *complete influence* method $\mathcal{I}^C(x)$:

$$err(\mathcal{I}, x) = d(\mathcal{I}(x), \mathcal{I}^C(x)) \tag{6.10}$$

For each instance of each dataset, we generate the error score of every method, allowing us to compare their performances across the different datasets we collected. Each error score is the distance of the method from the *complete* method. Thus, a lower error is indicative of a more precise estimation of the *complete* method.

Results and interpretation

Figure 6.7 indicates the computation time of all the explanation methods. This time takes into account every step of each method: the training of the models, the predictions necessary to calculate the influences, and the constitution of the correlated groups for the coalitional method. As expected, the *coalitionnal* influences are much more efficient than the *k-complete* influences (less than 200s for the first ones compared to 700s for the other ones). The decrease in computation time for 9 attributes is explained by the important decrease in the mean number of instances. This makes each retraining faster to do, even if there are twice more subgroups to take into account.

Figure 6.8 depicts the mean error score, aggregating the error score (Equation 6.10) of each explanation method for each of our 324 datasets. In this figure, the lowest curve is the closest to the *complete* influence method and thus is performing the best. As expected, the *linear* influence gives the worst results. This is explained by the fact it only considers single attributes, which is far from all the possible groups of attributes considered by the *complete* influence.

The comparison between the *k-complete* influences and the *coalitionnal* influences (represented by their alpha parameters) is more delicate. Certainly, the *k-complete* influences outperform the other influences, in the majority of the cases, but with a high cost in execution time. Also, it does not mean the *coalitionnal* influences are less interesting for our case. They generate a smaller subset of groups of attributes while preserving an acceptable error score. Overall, the *coalitional* methods are not as satisfactory as the *k-complete* in term of effectiveness. But their comparatively very low execution time makes them far more desirable when confronted with large datasets with an important number of attributes.

Considering these results, it seems the *k-complete* influence is preferable with a relatively small k and a dataset having few attributes, while the *coalitional* influence seems to become preferable with a higher number of attributes. Obviously, larger subgroups seem to

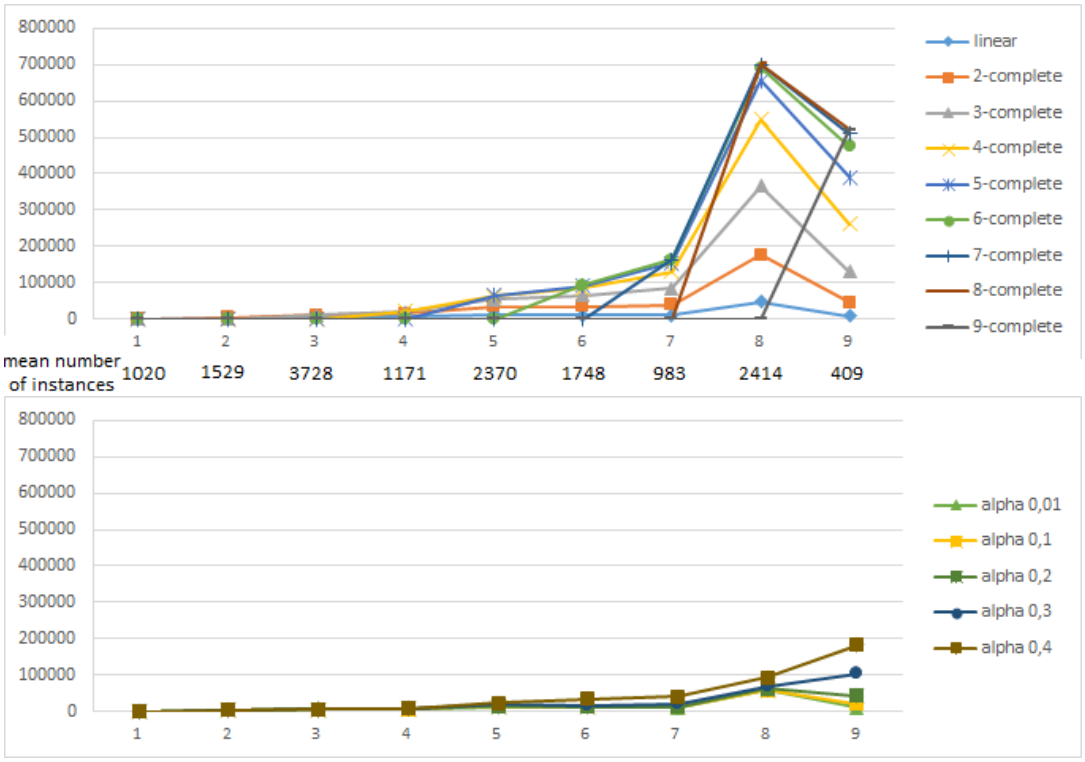


Figure 6.7: Execution time, in milliseconds, of each explanation method depending on the number of attributes in the dataset. The mean number of instances is added for comparison.

increase the precision of the methods, but in the case of the *coalitional* method, their impact on computation time seems to be relatively small when compared to the performance gain. Besides, our study about the groups generated by the grouping algorithm shows that the number of groups stays relatively small, even for large alphas. As an example, for the datasets of 9 attributes, the mean size of the biggest generated group is 4, using an alpha of 0.4. This means that the *coalitional* influence is working with far less information than the *k-complete* one. Studying the influence of different ways to generate the coalitions of attributes on the *coalitional* influence could be a good aim for the near future. With new methods, it could be possible to find more relevant groups, bettering the precision of the explanation without an important computational cost. Moreover, it would be interesting to investigate overlapping attribute coalitions, using algorithms allowing for attributes to be in different coalitions, enabling the exploration of more subgroups if necessary.

6.5 Improving on the coalitional influence

As we have seen, the coalitional influence seems to be a good candidate, but its reliance on a grouping algorithm to function calls for a deeper exploration of the performances

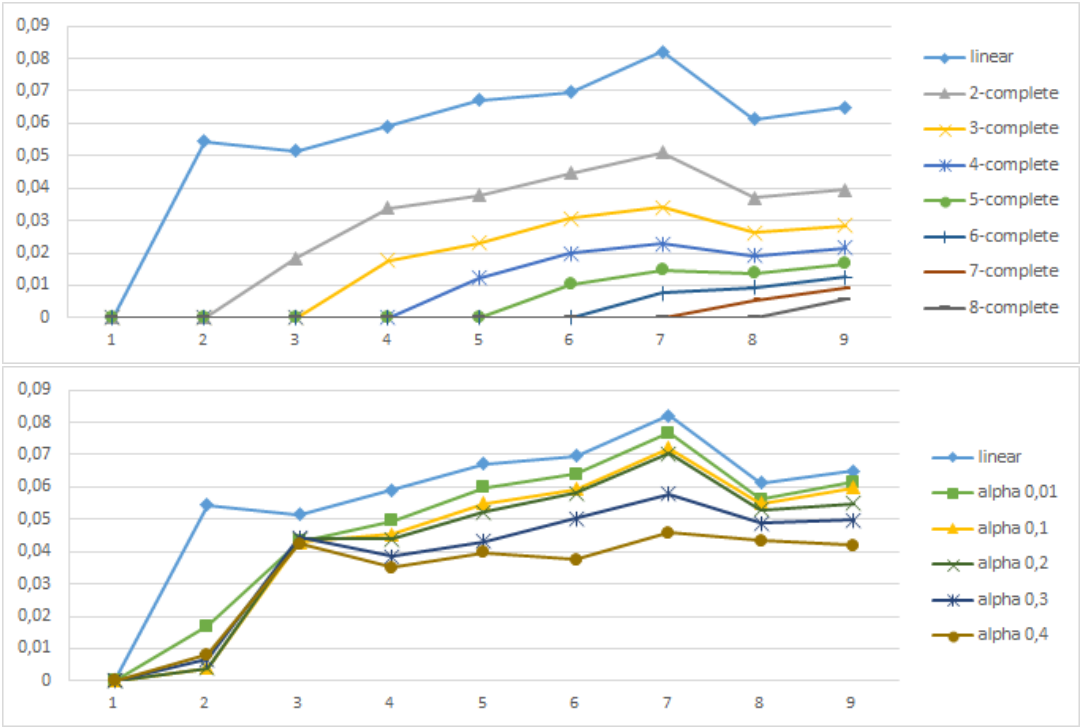


Figure 6.8: Error score between each explanation method and the *complete influence* depending on the number of attributes in the dataset.

of the *coalitionnal* influence, depending on the grouping algorithm used. We based our first algorithm on the work of [Hen14]. The new algorithms we want to test are based on the Variance Inflation Factor (VIF), the Spearman correlation factor, and the Principal Component Analysis (PCA) of a dataset. For each algorithm, we implement a parameter that controls the size of the subgroups that are generated. A higher value of this parameter generates larger groups whereas a smaller value produces smaller groups.

6.5.1 Principal component analysis based coalition

The main principle of a Principal Component Analysis is to reduce a dataset to its simplest expression in terms of attributes. In other words, if the dataset is considered a multidimensional matrix, the PCA aims to reduce its dimensionality as much as possible. To do that, the different attributes of the dataset are combined linearly, the result being a new set of attributes, each new attribute being a linear combination of the previous ones.

Our reasoning, for this approach, is to consider the set of combined attributes (summarized by the new attribute of the PCA) as a group of influence.

Given a dataset $D = (A, X)$ composed of a set of n attributes $A = \{a_1, ..., a_n\}$, and a set of instances X where $x \in X, x = \{x_1, ..., x_n\} \forall i \in [1..n], x_i \in a_i$.

We can apply a principal component analysis which produces a new dataset $D' =$

(A', X') such as $A' = \{a'_1, ..., a'_m\}$ with each new attribute being a linear composition of the previous attributes : $\forall i, a'_i \in A', \exists \{\alpha_1, ..., \alpha_n\} \in R^n, a'_i = \alpha_1 * a_1 + ... + \alpha_n + a_n$.

Each new instance is associated with an instance of the previous dataset. $\forall x' = \{x'_1, ..., x'_m\} \in X', \exists!\ x \in X, \forall i \in [1, .., m] \exists\ \alpha_1, ..., \alpha_n \in R^n, x'_i = \alpha_1 * x_1 + ... + \alpha_n + x_n$.

Given this set of factors $\alpha_1, ..., \alpha_n$, for each attribute, we consider each factor as an evaluation of the importance of the attributes in the group. We can then constitute a coalition of attributes by exploiting the groups formed by the most important factors. This gives us the algorithm 4. For the sake of simplicity, we consider each $a' \in A'$ as a vector of its α_i factors.

Algorithm 4 PCA-based coalition extraction.

Require: a threshold t and the set of attributes A' of the PCA

Ensure: σ a coalition of attributes

```
σ ← {}
for all a' ∈ A' do                          ▷ for each attribute generated by the PCA
    g ← {}                                  ▷ g, a new possible group
    αmax ← max(a' = α_1, ..., α_n)          ▷ find the most important factor
    for all α_i ∈ a' do
        if α_i ≥ α max * (1 − t) then
            add a_i to g                    ▷ the attribute is included in the group if close to the max
        end if
    end for
    add g to σ
end for
return σ
```

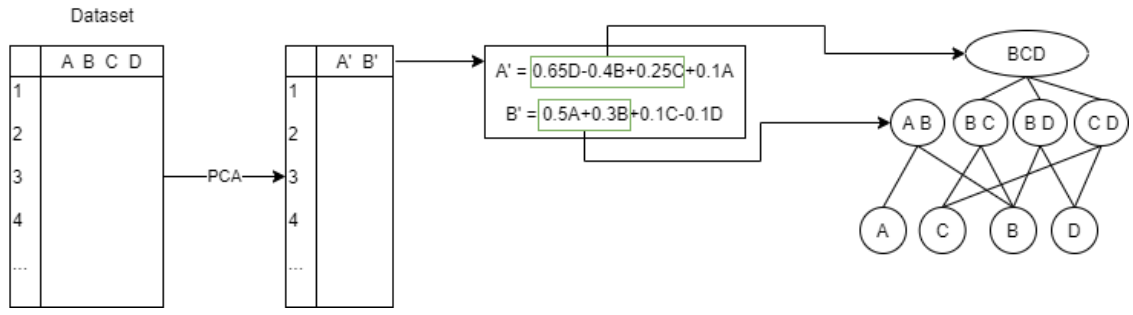


Figure 6.9: Depiction of the groups calculated by the PCA based coalition method for a 4 attributes dataset. The new attributes formed by the PCA are a combination of the previous attributes. The attributes with the highest coefficient for each new attribute are considered as part of a group to be calculated.

Example 15 *Given our previous dataset of 4 attributes, we run a PCA on it. The new attributes generated are as in Figure 6.9. We have two principal components :* $A' = 0.65D - 0.4B + 0.25C + 0.1A$ *and* $B' = 0.5A + 0.3B + 0.1C - 0.1D$*. Here, we consider each attribute with the highest associated coefficients as part of a group. So, here, we have two groups: one for* A'*,* $\{D, B, C\}$ *and one for* B'*,* $\{A, B\}$*. We then calculate the total influence of each attribute as the combined influences of each attribute alone and each subgroup of the two generated groups containing the attribute. Thus, the total influence of* A *is composed of the influence of* $\{A\}$ *and* $\{A, B\}$ *as no other group of subgroup generated contains* A*. For the total influence of* B*, we use the influence of* $\{B\}, \{A, B\}, \{B, C\}, \{B, D\}$ *and* $\{B, C, D\}$*. The total influence of* C *depends on the influences of* $\{C\}, \{B, C\}, \{C, D\}$ *and* $\{B, C, D\}$*. Finally, the influence of* D *is constituted by the influences of* $\{D\}, \{B, D\}, \{C, D\}$ *and* $\{B, C, D\}$*.*

6.5.2 Variance Inflation Factor and reverse Variance Inflation Factor-based coalition

The variance inflation factor (VIF) is an estimation of the multicollinearity of the attributes of the dataset regarding a given target attribute.

Given a dataset $D = (A, X)$, the VIF value of $a \in A$ is calculated by running a standard linear regression with a as the target for the prediction. Then, given R the coefficient of determination of the linear regression, we have:

$$VIF(a) = \frac{1}{1 - R^2} \tag{6.11}$$

It is commonly accepted that a variance inflation factor superior to 10 indicates strong multicollinearity of the attribute with other attributes of the dataset. This threshold of 10 is arbitrary but considered as a standard in numerous publications (e.g. [Mak19]). Moreover, when an attribute is removed from the dataset, the VIF of the attributes multicollinear with it decreases. Then, we can automatically detect groups of attributes by calculating the VIF of each attribute (considered as a target) of the dataset, and then comparing them with a new VIF calculation with an attribute removed. For this purpose, we consider two possible approaches:

- Considering as a priority the calculation of strongly multicollinear groups of attributes: Those are groups of attributes with a dependency on one another. In the context of this approach, attributes whose VIF varies strongly when an attribute is

removed from the dataset is considered as part of the group.

- Considering as a priority the calculation of weakly or non-multicollinear groups of attributes: Given the fact that correlated attributes tend to bring the same information to the model, it may be preferable to prioritize groups for which the addition or removal of an attribute changes greatly the information brought by the group.

Algorithm 5 VIF-based coalition extraction.

Require: a threshold t, the set of attributes of the dataset A and a function $VIF(A)$ calculating the array of all the VIF of all the subsets of a set of attributes

Ensure: σ a coalition of attributes

```
σ ← {}
oldvifs ← VIF(A)                    ▷ calculating the initial VIFs of the attributes
for all a ∈ A do
    g ← {}
    add a to g
    newvifs ← VIF(A/a)
    for all a' ∈ A do
        if newvifs(a') < oldvifs(a') * (0.4 + t) then
            add a' to g
        end if
    end for
    add g to σ
end for
return σ
```

These two approaches are named *VIF coalition* and *reverse VIF coalition*, respectively. This gives us the algorithm 5, for the *VIF coalition*. The *reverse VIF coalition* can be obtained simply by replacing the condition for adding an attribute to a group by $if\ newvifs(a') > oldvifs(a') * (1 - t * 0.05)$. This supplementary ratio of 0.05 has been obtained by preliminary experiments, which showed that just keeping the $1-t$ factor led to a generation of all the possible subgroups, which defeat the principle of an approximation.

Example 16 *Given our dataset of 4 attributes, we calculate the VIFs of each attribute. Then, we calculate the VIFs again each time with one of the attributes removed. The results are depicted in Figure 6.10. In the case where A is removed, we see that the VIFs of B and C vary greatly. Thus, our first group is* $\{A, B, C\}$*. Then, when B is removed, only*

the VIF of A varies a lot. We then have a second group: $\{A, B\}$. *Finally, we can see that removing* C *and* D *do not make the other VIFs vary in a significant way. Because of that, the attributes* C *and* D *are considered as singletons. As the group* $\{A, B\}$ *is contained by* $\{A, B, C\}$, *our final coalition is* $\{\{A, B, C\}\{D\}\}$. *The complete influence of* A *is constituted of the influences of* $\{A\}$, $\{A, B\}$, $\{A, C\}$ *and* $\{A, B, C\}$. *the complete influence of* B *includes* $\{B\}$, $\{A, B\}$, $\{B, C\}$, $\{A, B, C\}$. *The complete influence of* C *contains* $\{C\}$, $\{A, C\}$, $\{B, C\}$, $\{A, B, C\}$. *Finally, the complete influence of* D *only contains* $\{D\}$.

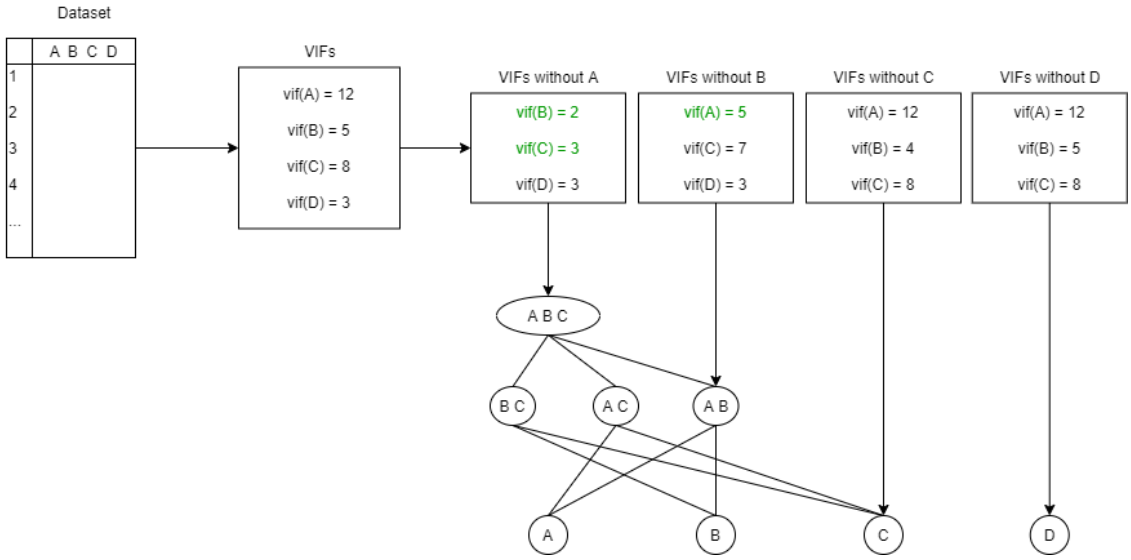


Figure 6.10: Depiction of the groups calculated by the VIF based coalition method for a 4 attributes dataset. the VIFs are calculated for each attribute and are recalculated with an attribute absent from the dataset. The attributes whose VIFs varies the most are considered as grouped with the removed attribute. If no VIF is changed, the removed attribute is considered as a singleton.

6.5.3 Spearman and reverse Spearman correlation-based coalition

A limit of the variance inflation factor is the sole consideration of multicollinearity, while a correlation between attributes might not be linear. This problem is addressed through the Spearman correlation coefficient, which takes into account non-linear correlations. Spearman being not multicollinear, the calculation of the correlation between attributes has to be done by pairs. Thus, the method consists of generating the matrix of all the correlations of each pair and then deciding which attributes are part of a group. For this method, we have the same two possibilities as for the *VIF* method: we can either prioritize the calculation of strongly correlated attributes or on the contrary, prioritize groups of non-correlated attributes. These two approaches are named respectively *Spearman coalition* and *reverse Spearman coalition.*

Given a dataset $D = (A, X)$, with $A = \{a_1, ..., a_n\}$ the correlation matrix C is obtained by computing the Spearman correlation coefficient of each attribute couple : $C(1, 2) = corr(a_1, a_2)$. Thus C is symmetrical and have 1 as the value of its whole diagonal. For each line i of the matrix C, we consider as grouped with a_i the attributes strongly (or weakly) correlated with a_i, for the *Spearman coalition* (or the *reverse Spearman coalition*).

Algorithm 6 Spearman-based coalition extraction.

Require: a threshold t, the set of attributes of the dataset A, and a function $spearman(A)$ calculating the matrix of all the absolute Spearman correlation coefficient of all the subsets of a set of attributes. a max and min functions which returns the maximum and minimum of a matrix line.

Ensure: σ a coalition of attributes

$\sigma \leftarrow \{\}$

$corrmat \leftarrow spearman(A)$ ▷ calculating the correlation matrix

for all $a \in A$ **do**

$g \leftarrow \{\}$

for all $a \in A$ **do**

if $corrmat(a, a') > max(corrmat(a)) * (1 - t)$ and $max(corrmat(a)) > 0.1$ **then**

▷ If the most correlated attribute have a coefficient less than 0.1, we consider a as a singleton

add a' to g

end if

end for

add g to σ

end for

return σ

The algorithm 6 details the *Spearman coalition* method. The *reverse Spearman coalition* method can be obtained by replacing the condition for adding an attribute to a group by $corrmat(a, a') < min(corrmat(a)) + max(corrmat(a)) * t$ and $min(corrmat(a)) < 0.5$. This adds the least correlated attributes up to a threshold : if the attribute least correlated to a have its Spearman correlation to a superior to 0.5, we consider the attribute a as a singleton.

Example 17 *Given our previous dataset of 4 attributes, we calculate the matrix of the spearman correlation coefficients as depicted in Figure 6.11. In this matrix, we iterate on each row of the matrix, in order to create groups based on the most correlated attributes.*

In the first line, we see that the attribute most correlated to A is B, and the two other attributes are very weakly correlated to A. Thus, we have a first group: $\{A, B\}$*. The second line tells us that A and C are both strongly correlated to B. So, we have a second group:* $\{A, B, C\}$*. Similarly, the third line indicates that B and D are correlated to C, and so we add a third group:* $\{B, C, D\}$*. Finally, by looking at the last line we learn that only C is strongly correlated to D and so our last group is* $\{C, D\}$*. As the two groups of cardinal 2 are contained by the two groups of cardinal 3, we have our final coalitions: {{A,B,C},{B,C,D}}. With this coalition, the complete influence of the attribute A is composed of* $\{A\}$*,* $\{A, B\}$*,* $\{A, C\}$ *and* $\{A, B, C\}$*. B is composed of* $\{B\}$*,* $\{B, D\}$*,* $\{A, B\}$*,* $\{B, C\}$*,* $\{A, B, C\}$ *and* $\{B, C, D\}$*. Finally, the complete influence of D is composed of* $\{D\}$*,* $\{C, D\}$*,* $\{B, D\}$ *and* $\{B, C, D\}$*.*

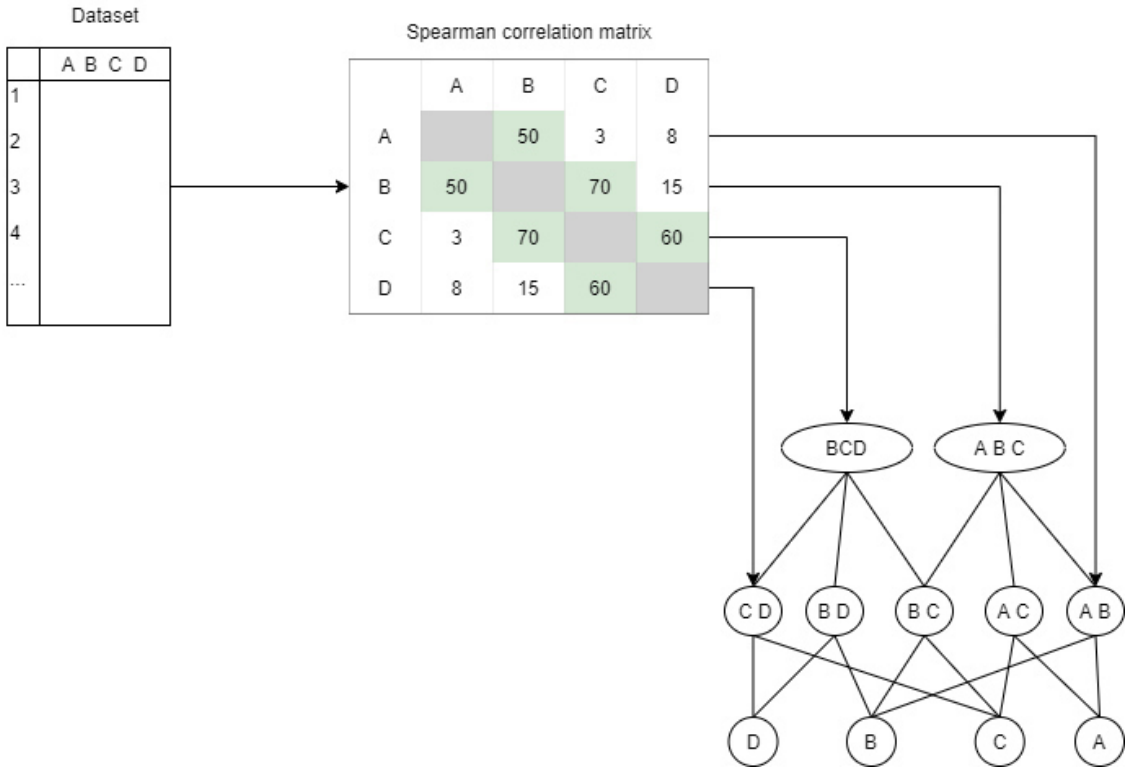


Figure 6.11: Depiction of the groups calculated by the spearman based coalition method for a 4 attributes dataset. The spearman correlation matrix is calculated. For each line, the attributes most correlated with the line's attribute are considered as part of a group.

6.6 Experiments : comparing the different simplification methods

In this section we aim to evaluate the performances of each coalition calculation method, considering their precision when compared to the *complete* influence and their computa-

number of attributes	1	2	3	4	5	6	7	8	9
Mean number of instances	1020	1529	3728	1171	2370	1748	983	2414	409

Table 6.1: mean number of instances for datasets with a given number of attributes.

tional time. We also give an overview of the group characterization for each coalition method.

- **Experimental protocol**

This experiment is realized using the same hardware setup and the same datasets and workflows as in Section 6.4.3. For each instances of the datasets, we generate the *complete* influence for the baseline, along with the *coalitional* influence. The *coalitional* influences are generated using the different group generation methods described in Section 6.5, We also use the same Euclidean distance and alpha parameter as in 6.4.3

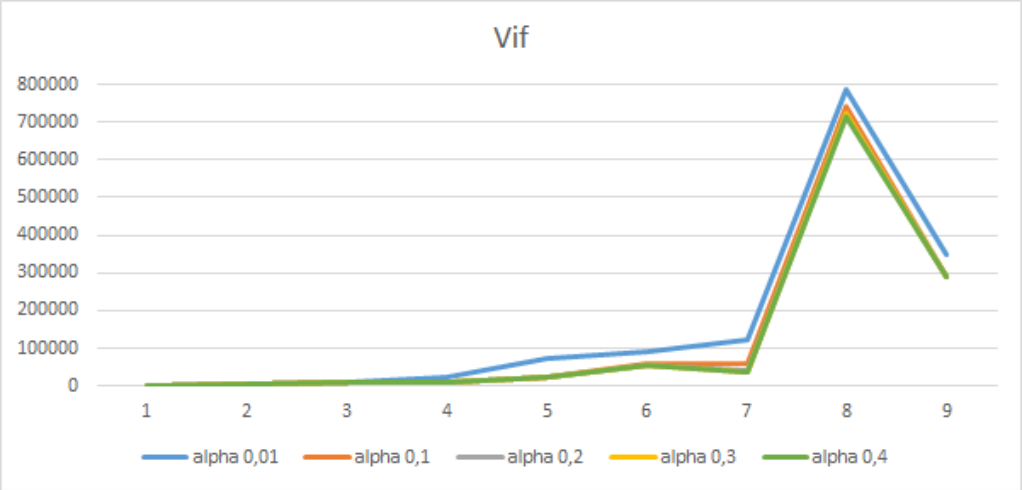
6.6.1 Calculation time and Error scores

Figures 6.12 and 6.13 give the performance and computational time in miliseconds of each coalitional method, respectively (for different values of their threshold parameter).

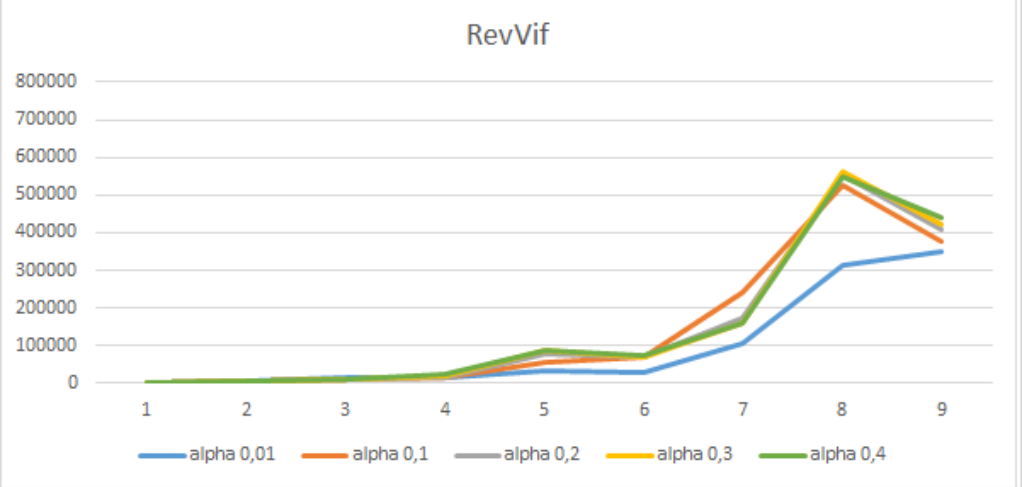
For readability, Table 6.1 details the mean number of instances for each number of attributes. This can have an impact on computation time and explains the variations of Figure 6.12. This figure includes the computation time for generating the groups of attributes and for explaining each instance of the dataset. The decrease of the computation time for the case of 9 attributes is explained by the important decrease in the mean number of instances. This makes each retraining faster to do, even if there are potentially twice more subgroups to take into account.

Figure 6.13 depicts the mean error score, aggregating the error score (Equation 6.10) of each explanation method for each of our 324 datasets. In this figure, the closer the curve is to 0, the closer it is to the *complete* influence method.

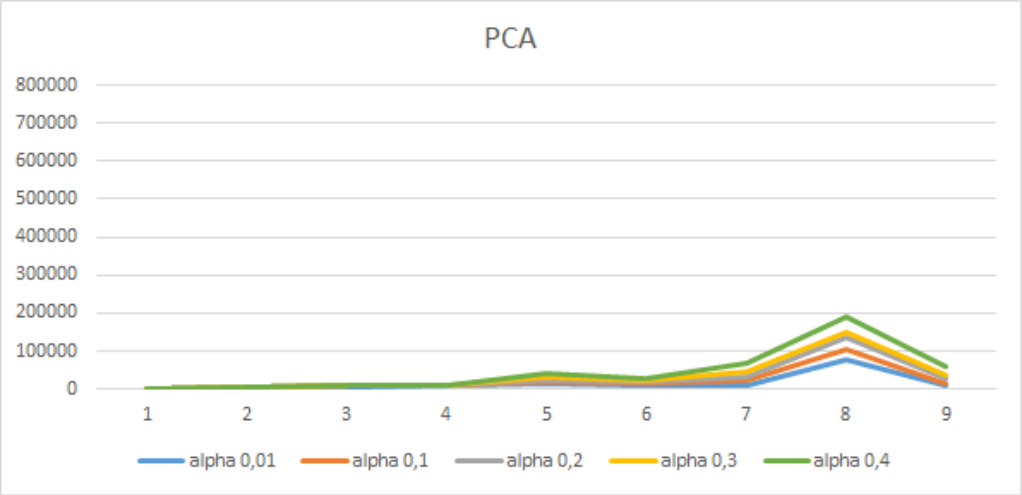
As we can see, in an overall analysis, the *VIF* method seems to be the worst, with poor performance and a long computational time. This can be explained by the fact that the attributes of the generated group are correlated to one another, which means that the information brought by these groups and subgroups is very redundant. We can suppose a lot of groups are calculated (see Section 6.6.2 for more details), but they often bring nearly the same information each. *Spearman* has a far better computation time than *VIF* but still



(a) Time values for VIF based coalitional explanation.

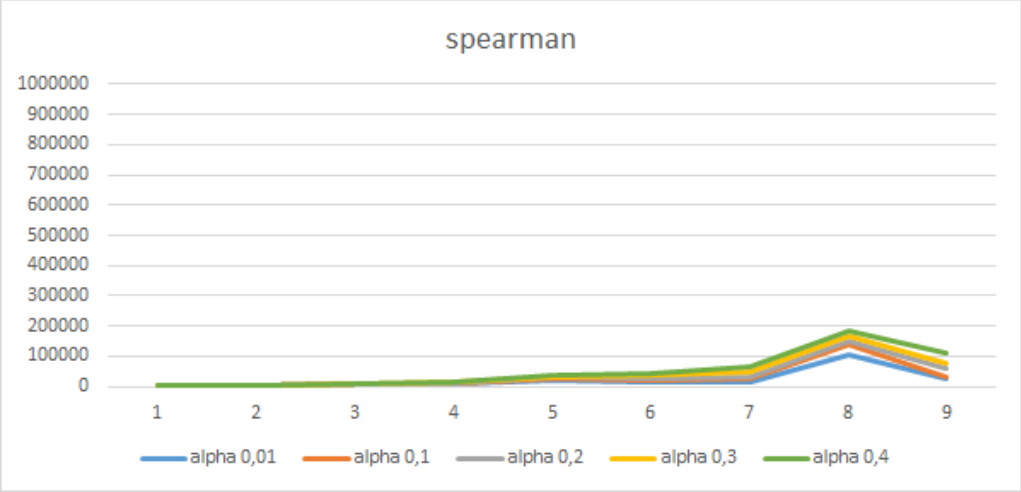


(b) Time values for Reverse VIF based coalitional explanation.

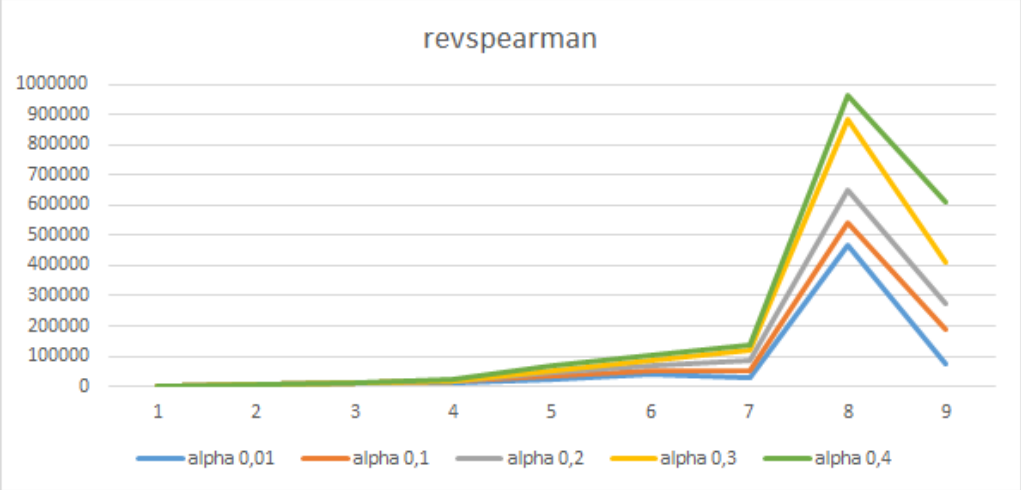


(c) Time values for PCA based coalitional explanation.

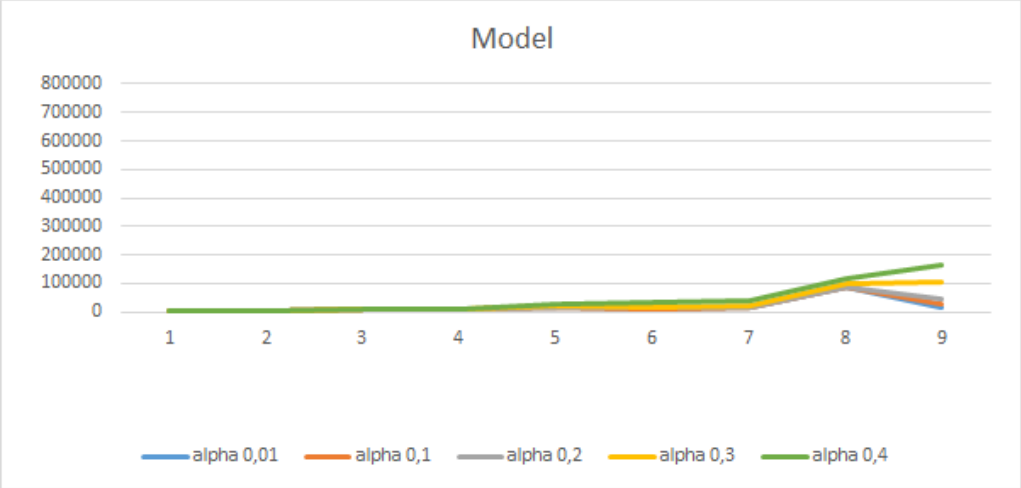
Figure 6.12: Calculation time of each coalitional method versus the number of attributes in the dataset.



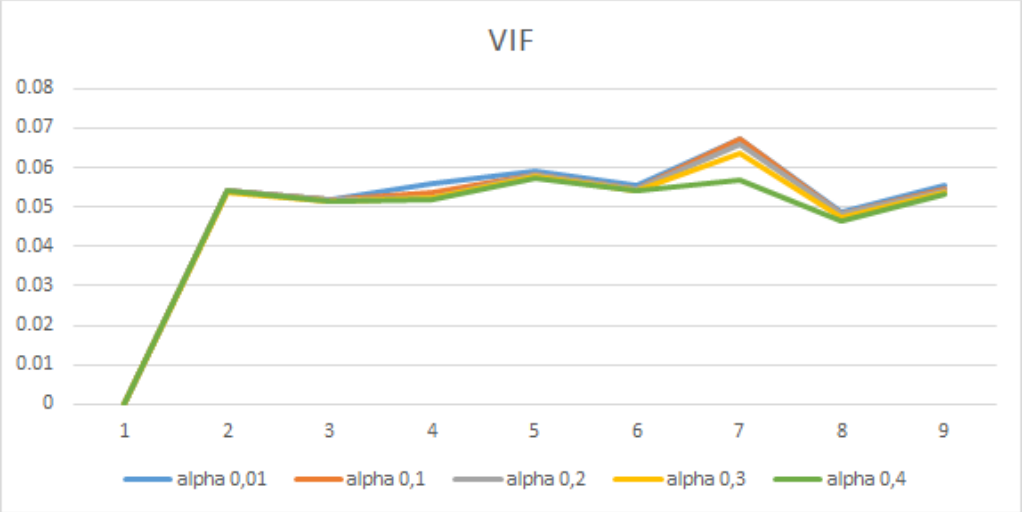
(d) Time values for Spearman based coalitional explanation.



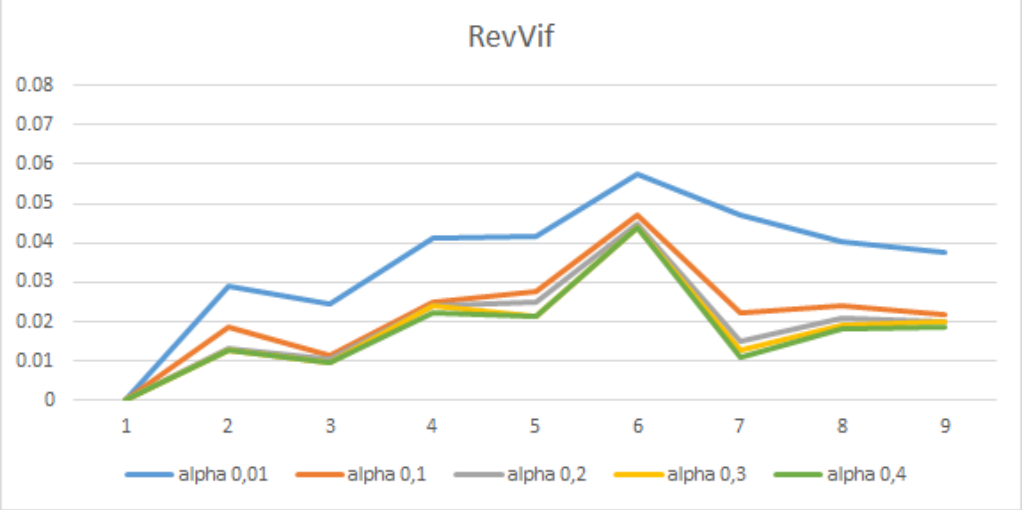
(e) Time values for Reverse Spearman based coalitional explanation.



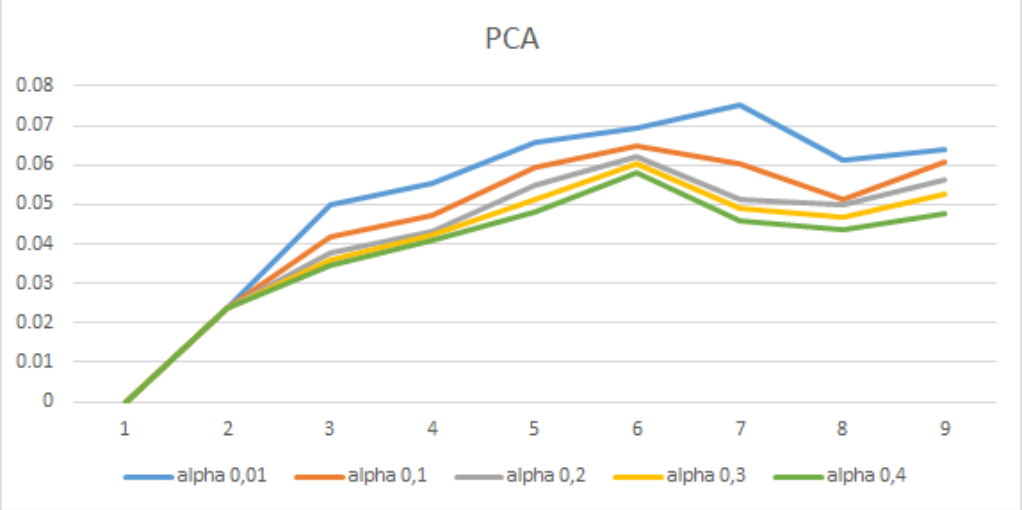
(f) Time values for PCA based coalitional explanation.



(a) Error values for VIF based coalitional explanation.

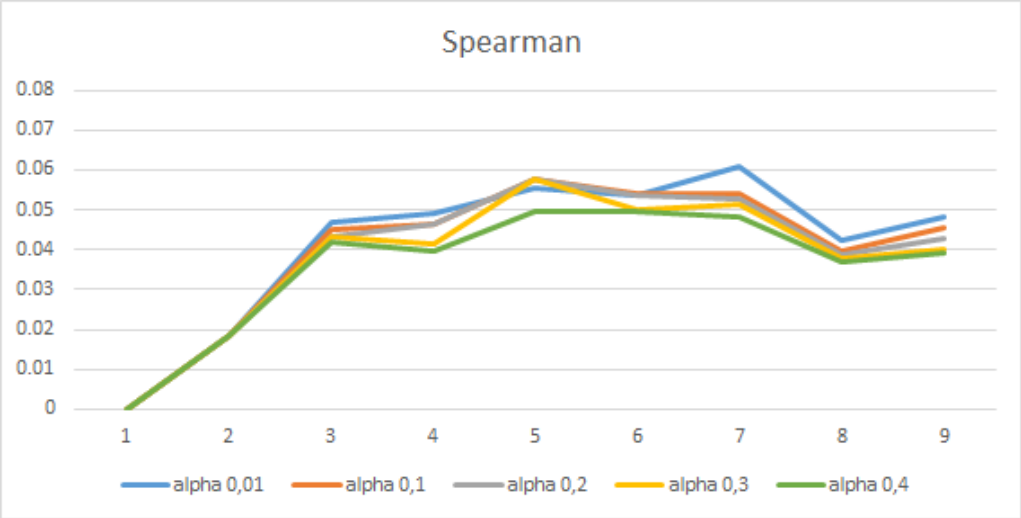


(b) Error values for Reverse VIF based coalitional explanation.

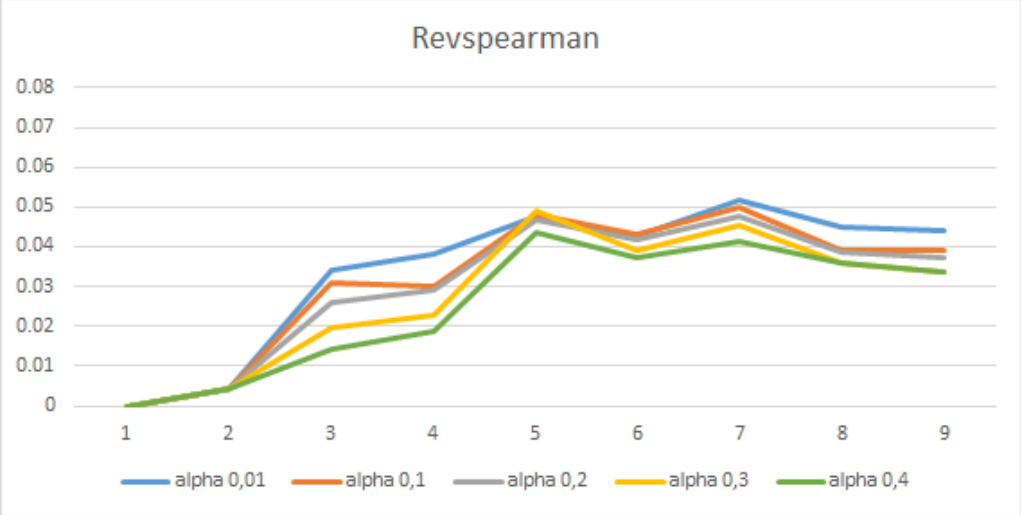


(c) Error values for PCA based coalitional explanation.

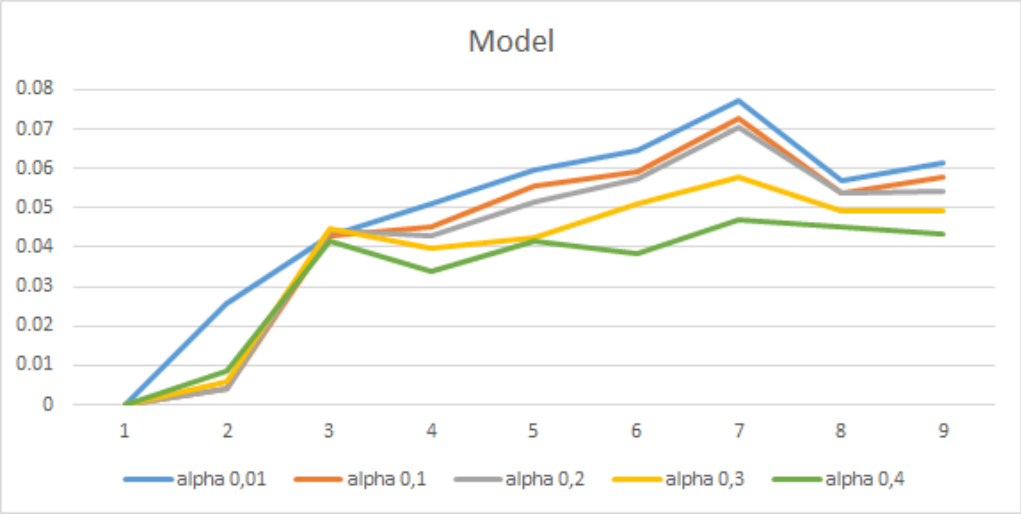
Figure 6.13: Error score between each coalitional method and the complete influence, versus the number of attributes in the dataset.



(d) Error values for Spearman based coalitional explanation.



(e) Error values for Reverse Spearman based coalitional explanation.



(f) Error values for PCA based coalitional explanation.

has a poor performance overall, probably for the same reasons. As an example, *PCA* has a better performance but a computation time very similar to *Spearman*. *RevSpearman* has an overall better performance than part of other methods, but this performance is paid by the longest computation time, without reaching the best performance. This can be explained by the group calculation method, which does not take into account the possible correlation of two attributes that are both not correlated with the original attribute of the group (no transitivity). This leads, as for *VIF* and *Spearman*, to the calculation of redundant information, which increases the computation time without improving much the performance. The *PCA*, *RevVIF*, and *Model* methods each seem to have their strong and weak points. The *RevVIF* is more precise than the other two but at a cost of greatly increased computation time. Instead of focusing on the correlated groups, the *RevVIF* method relies on the least correlated, thus a greater diversity of information is taken into account. While the *Model* and *PCA* methods are less exhaustive in their approaches, they seem to have a far lower computational time, the evolution of computation time against the number of attributes being far less steep than for *RevVIF*. by comparison with the calculation times of the *k-complete* method, only the *Vif* and *RevSpearman* methods have a computation time similar to it. Yet, the *k-complete* method has a far better performance than those two. In fact, no *coalitional* method achieve a better performance than the *K-complete* method, but the *RevVif* and *RevSpearman* succeed in achieving a similar performance for middle levels of k and alpha, and *RevVif* does it while maintaining a shorter computation time than the k-complete.

6.6.2 Group characterisation

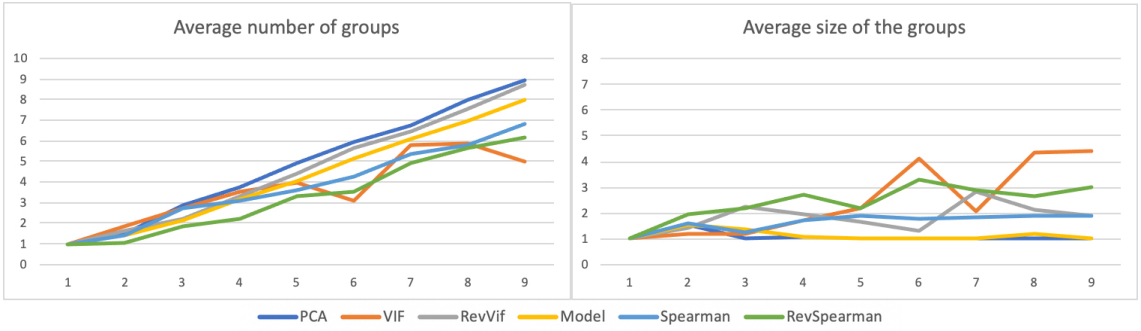


Figure 6.14: Group characterisation with alpha = 0.01.

Figures 6.14 and 6.15 compare the average number and average size of the groups of attributes generated by each coalitional method, respectively (for the two ends alpha = 0.01 and 0.4).

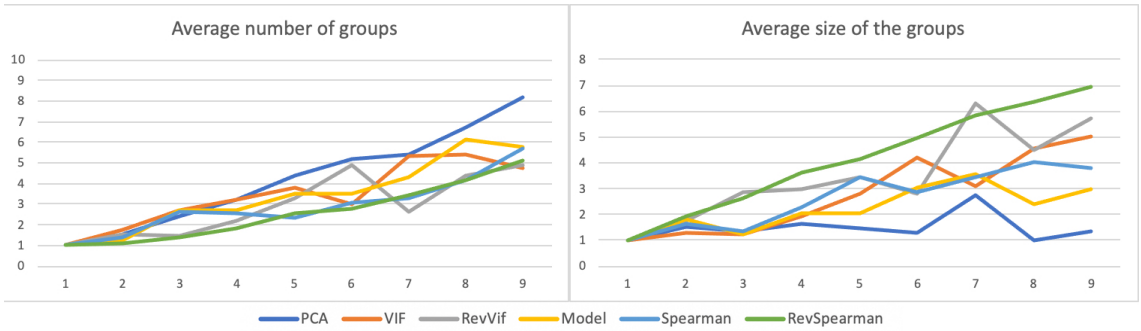


Figure 6.15: Group characterisation with alpha = 0.4.

We can note that *RevVIF*, *RevSpearman*, and *VIF* are the three methods generating the highest average group sizes, compared to the other methods. This phenomenon can explain why these methods minimize best the error scores as discussed in Section 6.6.1. Indeed, the larger the groups are, the more exhaustive they are in terms of coalition influence that can correctly explain an instance for a predictive model. More surprisingly, the high average number of groups seem not to induce a good error score. For example, the *RevSpearman* method generates, for the two alpha thresholds, the lowest number of groups, for most of the cases, whereas its error rate is one of the lowest. This can be explained by the generation of a lot of small groups (singletons or couples), rather than a few large ones. After all, the *complete influence* is the equivalent of the *coalitional influence* using a single group containing all the attributes.

6.7 Conclusion and perspectives

In order to tackle the issue of **helping a non-expert user understand and analyze the results produced by a data analysis process**, we first have built a basis on which to develop a prediction explanation system, opting for the *additive* method. This basis led us to the conclusion that the *complete* method is preferable for our goal. We then concluded that this method has a far too long computation time, bringing us to the problem of simplifying this method. We then developed and compared a large number of potential simplified approaches : The *k-complete* method and the *coalitional* method, with its diverse attributes grouping methods (*model*, *PCA*, *VIF* and *Spearman* coalition methods). This experiment leads us to conclude that some of these methods are promising, depending on the more important criterion. If raw performance is the main concern for deciding which method to use, the *k-complete* method seems to be the best candidate. Yet, its cost in computation time can lead to a more nuanced choice, with the *RevVif*, *PCA* or *Model* based *coalitional* methods. These methods have a varying degree of performance,

all with a different computation time, with room for a wide range of possible compromises between performance and speed. These compromises bring more flexibility to the methods, depending not only on the choice of the user but also on the size of the dataset being studied.

Concerning future potential researches, there are even more possible ways of simplifying the complete method to study. Moreover, the subject of finding solutions to apply these methods on larger datasets is very interesting. Although our simplified methods are faster than the complete method, they still have a limit to the size of the datasets they can be applied to. As an example, given a dataset with a large number of (let's say 300 attributes), our methods would still take hours to complete. As even the *single* method could not achieve a prediction explanation in a reasonable time given such a dataset, how could we still evaluate the importance of each attribute? A possible answer would be to run a global influence study first and collect the ten most globally important attributes. Then, the predictions explanations for single instances could be run on only those ten most important attributes to keep a reasonable computation time. With ten attributes, we avoid the calculatory explosion caused by a too large number of attributes. By limiting the number of attributes in the explanation, we also avoid overwhelming the user with too much information, which would make our explanation hard to interpret.

Finally, an interesting perspective is a study on which instances to select for presenting to the user. Which set of instances prediction explanation gives the most accurate picture of the model being studied? [RSG16] proposed a first solution to this problem, but there is no doubt there are many other possible ways to select instances for presentation, and evaluate the quality of the selection of those instances. As an example, the instances could be selected in a way that shows the data points where the model's behavior varies the most. Another possible approach would be to select points of the dataset where the model has the most trouble to classify the instances.

In chapter 7 these methods are studied in the larger scope of an actual application, allowing us to research the potential applications of prediction explanation in the context of user assistance.

References

[CMB18] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. "Visualizing the feature importance for black box models". In: *Joint European Conference on*

Machine Learning and Knowledge Discovery in Databases. Springer. 2018, pp. 655–670.

[Hal09] Mark Hall et al. "The WEKA data mining software: an update". In: *ACM SIGKDD explorations newsletter* 11.1 (2009). Publisher: ACM, pp. 10–18.

[Hen14] Andreas Henelius et al. "A peek into the black box : exploring classifiers by randomization". In: *Data mining and knowledge discovery* 28.5-6 (2014). Publisher: Stockholms universitet, Institutionen för data- och systemvetenskap, pp. 1503–1529.

[LL17a] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4765–4774.

[LL17b] Scott M Lundberg and Su-In Lee. "Consistent feature attribution for tree ensembles". In: *arXiv preprint arXiv:1706.06060* (2017).

[Mak19] Sara Makki. "An Efficient Classification Model for Analyzing Skewed Data to Detect Frauds in the Financial Sector". PhD thesis. Université de Lyon; Université libanaise, 2019.

[Qui86] J.R. Quinlan. "Induction of Decision Trees". In: *Machine Learning* 1.1 (Mar. 1986), pp. 81–106.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. event-place: San Francisco, California, USA. New York, NY, USA: ACM, 2016, pp. 1135–1144.

[Sha53] L. S. Shapley. "A value for n-person games". In: *Contributions to the Theory of Games* 28 (1953). Publisher: Princeton University Press, pp. 307–317.

[ŠK08a] Erik Štrumbelj and Igor Kononenko. "Towards a Model Independent Method for Explaining Classification for Individual Instances". In: *Data Warehousing and Knowledge Discovery*. Ed. by Il-Yeol Song, Johann Eder, and Tho Manh Nguyen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 273–282.

[SK10] Erik Strumbelj and Igor Kononenko. "An Efficient Explanation of Individual Classifications Using Game Theory". In: *J. Mach. Learn. Res.* 11 (Mar. 2010). Publisher: JMLR.org, pp. 1–18.

[Van13] Joaquin Vanschoren et al. “OpenML: Networked Science in Machine Learning”. In: *SIGKDD Explorations* 15.2 (2013). Place: New York, NY, USA Publisher: ACM, pp. 49–60. DOI: 10.1145/2641190.2641198.

There is a rumour going around that I have found God. I think this is unlikely because I have enough difficulty finding my keys, and there is empirical evidence that they exist.

Terry Pratchett

Yes, linear algebra is elegant, but it is far more enjoyable when you are not the one doing it

---

William Raynaut

Chapter 7

A framework for assistance in data analysis

7.1 Introduction

In chapter 5, we have established a recommendation system, which recommends possible effective workflows to a user from his dataset and his preferences. In chapter 6 we created a prediction explanation system bringing a way to understand how a predictive model works. We now have a new problem to face: How can we bring our propositions together to answer to our issue of **helping a user who is not an expert in data analysis at building his own data analysis model**? To achieve our goal, we have to do more than just present the user with the recommendation and provide him with prediction explanations when he is seeing predictions from his model. We have to exploit the knowledge of the user of his own domain in order to guide him through data analysis. For this, we need to cleverly use our propositions in order to allow the user to see the prediction explanation not only as a visual aid but as a guide into how the models work and how they can be ameliorated. Our intuitions for this framework are that we can use the prediction explanation to facilitate three main tasks :

- Model selection: Choosing a model through their prediction explanation
- Attribute selection: Determining the interest of an attribute from its usage of the models
- Model exploitation: Using a model and its results with the aid of their explanation.

From these intuitions, we build a basic framework exploiting our tools, which is then developed in a functioning prototype for further experimentation. During this chapter, we

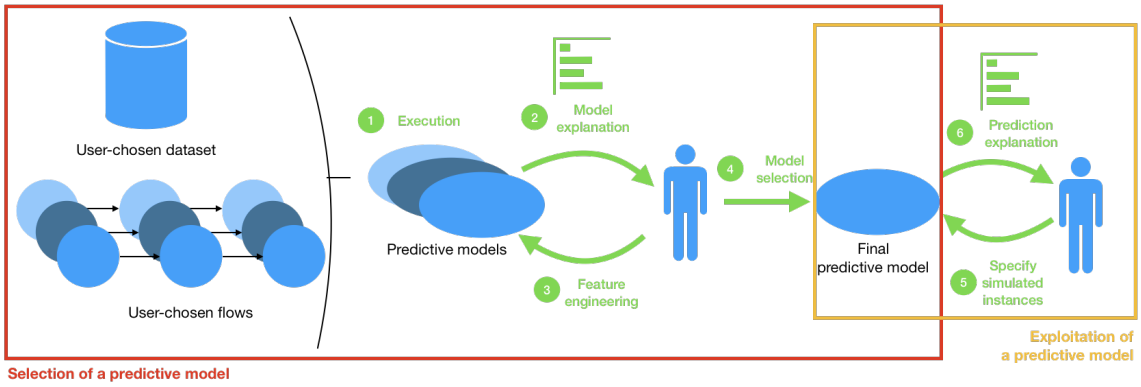


Figure 7.1: Building a predictive model.

first develop the general layout of our framework and detail its functionalities in Section 7.2. Then, in Section 7.3 we compare the functionalities of the framework with the objectives we have set for it, verifying our answers to our issues. Finally, in Section 7.4, we develop a prototype of our system and an experimental protocol, which would have been used for testing our entire proposal, but did not because of the ongoing pandemic.

7.2 Organisation of the framework

We now present the framework including the proposals described in the two previous chapters. Our framework is separated into three successive use cases. In Section 7.2.1, we show how a domain expert user can be guided through the complex process of selecting a predictive model among a set of possible ones. During this selection, the user is also guided through the process of selecting features to remove from his dataset if necessary. Once his model has been created, the process of exploiting it through new instances is described in 7.2.2, which illustrates how explanations bring new insights while using a predictive model.

Remember that this framework is intended for users who have no prior knowledge of machine learning, but who have expertise in their field (e.g. biologists, doctors, engineers...). These users produce data that they are required to analyze. It is therefore assumed that they have a solid knowledge of this data, but not of the machine learning methods.

7.2.1 Model selection via prediction explanation

The first principal function of our framework is depicted in the red square in Figure 7.1. It consists of five steps, which guide the user through the selection of a model by using prediction explanation, along with performing features selection.

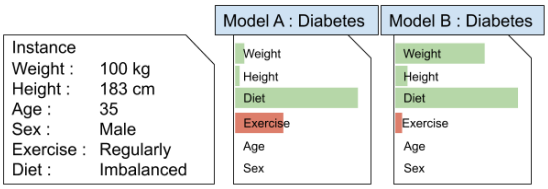


Figure 7.2: Comparing two prediction explanations.

1. *Workflow recommendation* - First, a user produces data he wants to analyze. The data is given as input of the recommender system, along with their specifications for the analysis: the target feature and their preferences in terms of results. The system then suggests a set of possible workflows that are the ablest to analyze the user's according to our recommendation system described in Chapter 5.

2. *Execution* - Among this selection of possible workflows, the user can select all or a set of them. He can access a description of each workflow and its inner working if desired, allowing him to perform a first selection of the possible workflows. The workflows selected are then executed and produce a set of predictive models trained on the user's dataset.

3. *Model explanation* - Using these models, the system can generate the classification of a given instance of the dataset and provide its afferent explanation for each model. These explanations take the form of attribute influences as described in chapter 6. For instance, in Figure 7.2, a user is informed that a particular patient is predicted to have diabetes by both models A and B, but that A made this decision considering mostly the patient's diet, while B also considers his weight as important. To help the user intuitively explore the models, a set of 10 instances are recommended for his review. This selection of instances aims to provide the user with a set of prediction explanations as diverse as possible, without overwhelming him with a space too large to be explored efficiently by humans.

4. *Feature engineering* - Thanks to the prediction explanations, a user can access the reasoning behind each model, allowing him to detect possible flaws in the proposed models. As an example, with the help of prediction explanation, personnel of a hospital performing a medical study described in [RSG16] realized that some attributes should not have been included in their dataset. Moreover, based on his domain of expertise, a user can assess the importance of each feature, compared to the importance given to them by the models. Thus, the user can select undesirable features and remove them from the dataset.

5. *Model selection* - Once the final desired features have been determined, the user exploits his domain knowledge to assess the reasoning behind each model. This assessment is based both on a global evaluation, such as Cohen kappa or the area under the ROC curve, and local information on the prediction. Then the user selects the desired final model by choosing the best performing model, but also the one with the most relevant use of the dataset features.

Along with this selection, the user could want to make his dataset better by selecting the truly interesting features and leaving out the ones less interesting to him. With these steps, the user selects a workflow relevant to his problem by evaluating it through its use of his data.

As the user knows his data characteristics, he can judge the relevance of each workflow interpretation of the data. Moreover, he is also able to spot eventual errors in his dataset, which are not always obvious for users not an expert in machine learning. As an example, adding a patient ID in a medical dataset is common in medical research, but is a mistake when trying to analyze it automatically. This problem is not evident for a non-expert, but by seeing that the produced models use the patients' IDs while analyzing the dataset, the user realizes that mistake and can correct it.

Another possible application is the choice of relevant workflows according to the attributes the user wants to see studied. In the situation where the user knows which attributes are most relevant to him, this prediction explanation during workflow selection is a great opportunity for him to choose the workflow most relevant to his interests.

Finally, thanks to the prediction explanations, the user directly sees the consequences the modifications done on the dataset have on the different workflows. This permits a finer building of the dataset when deciding which attributes to add or remove.

7.2.2 Exploiting a model via prediction explanation

Once a model has been trained on a dataset, the user wants to use it on new data. This is the second principal function of our framework, and it is depicted in the orange square in Figure 7.1.

5 Once a user has selected his preferred model, it is trained on his dataset. With the new predictive model ready, the user can create new instances and input them into the model to obtain predictions for these instances. Those predictions are produced along with their explanation, which indicates to the user how the attributes have been used for the prediction.

6 The user can then use those functionalities to explore the possibilities of predictions for different cases. He can then randomize given values of an instance or test hypotheses by modifying the values of the instances, which gives him new insight into his data.

In this sandbox context, the user can test different cases, real or hypothetical. The predictions for those new instances give the user new insight into the behavior of the class depending on the data. As an example, a doctor could test the evolution of chances for developing diabetes if the patient changes his diet or his physical activity. Moreover, by fixing a precise attribute and randomizing the others to generate a new set of instances, the user sees the effects of this particular value of the attribute on a larger scale. This functionality aims to give a wide array of possibilities to the user while assisting his interpretation of the obtained results through the help of prediction explanation.

7.3 Validation of the framework

To validate the answers to our original questions indicated in Section 7.1, we propose a mock-up of our framework. This mock-up illustrates a machine learning task, based on the well-known UCI Pimas Indians diabetes dataset (available on many platforms, as Kaggle[1]), since familiarity with the dataset is beneficial to the understanding of this validation.

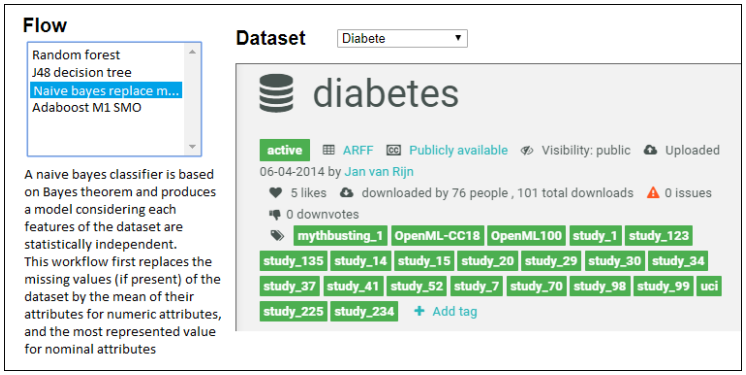


Figure 7.3: Workflow recommendation.

In our use case, a biologist is aiming to study the dataset of Pimas Indians diabetes and use our recommendation system to provide possible analysis workflows. First, as described in Section 7.2, the user enters the diabetes dataset as the input of the recommender system and asks it to perform a recommendation.

[1] https://www.kaggle.com/uciml/pima-indians-diabetes-database

7.3.1 Appropriate the results of the recommendation by yourself

Instead of recommending one of the workflows of the Pareto front, the user is presented with a set of the best recommendations from the Pareto front. A description of each workflow is made available to the user, so he can perform the first selection among the different options. Although these descriptions are necessarily technical, they are essential for a user to understand what is happening when each workflow is executed. The workflows and their descriptions are depicted in Figure 7.3. As an example, we can see in the figure that a workflow is not only the production of a predictive model but also successive operations of transformation applied in the dataset. These workflows are then executed and presented to the user through a set of selected instances. These instances are selected in a way that favors a large diversity in prediction explanations. The exact algorithm used here is the one presented in [RSG16]. The user can thus explore each predictive model through this set of instances, by viewing a diverse set of keypoints, illustrating the models. He then infers how the whole model works, with minimal information. The instances and their attached prediction explanations are depicted as in Figure 7.4. On the left, the user can select the instance he wants to study, and decide to eventually remove attributes from the dataset. On the right is presented the prediction explanation of the selected instance for each of the models (as the one surrounded in green). As an example, random forest and bagging J48 (an optimized decision tree) models mainly base their prediction on their blood pressure and age, whereas the naive Bayes is mostly influenced by the mass of the instance. This gives immediate access to the inner workings of each presented workflow, which is solely based on the domain knowledge of the user. Thus, by presenting the results and how they were obtained, the user is informed of the conclusion of the prediction, without having to rely blindly on the model. In our use case, we can see the scientist selects instance 49.

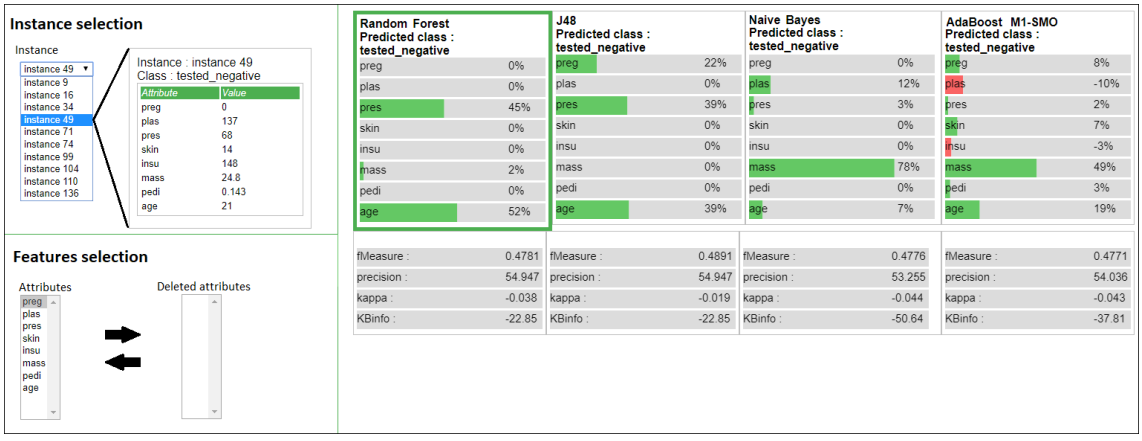


Figure 7.4: Visualization of prediction results through prediction explanations.

Therefore, through prediction explanation, the user can access a new type of information that does not rely on expertise in data analysis to be understood. He can understand and appropriate the results of the recommendation system thanks to his own domain-based knowledge: without understanding the inner workings of each model. He can visualize how each model uses the data to make predictions.

7.3.2 Giving user confidence in the produced results

Through this explanation method, the user can choose between models without having to rely solely on global measures of performance. He can use his own judgment rather than have his hand forced by the only proposal of a fully automated process. This also makes it possible to evaluate possible defects in the models, which is not always possible with only conventional metrics. As an example, the global accuracy of a model or the Kappa score does not "warn" a user of an inappropriate attribute that should be removed from the dataset.

In our mock-up example, the user can decide that the age of a patient is not that important in determining if he is likely to have diabetes. At the same time, if our user considers a patient's mass as a valid indicator, it indicates that the naive Bayes model is more interesting in his case (supposing the instances he reviewed are consistent with this explanation). This understanding of a model, its strengths, and its flaws give the user stronger confidence in what is being accomplished during the data analysis process. By pinpointing eventual problems in the predictive model, he also becomes able to know when the model is reliable.

7.3.3 Personalising a model without requiring data analysis knowledge

Once the user has studied his models, he can assess which workflows fit best his requirements. In particular, the user can identify which features are mainly used by the workflows, and decide which are important for his study. In our mock-up example, the biologist might want to study the impact of less evident diabetes indicators and decide to remove the insulin and plasma features from his dataset (as presented in Figure 7.5). This forces the workflows to use the other features, and maybe highlight new important characteristics of the dataset. We can see in Figure 7.5 that the J48 workflow has significantly changed its behavior, while the Adaboost model has simply adjusted the importance of each attribute.

By this process, the user accomplishes feature selection without having data analysis knowledge or expertise. His domain knowledge makes him capable of assessing the interest

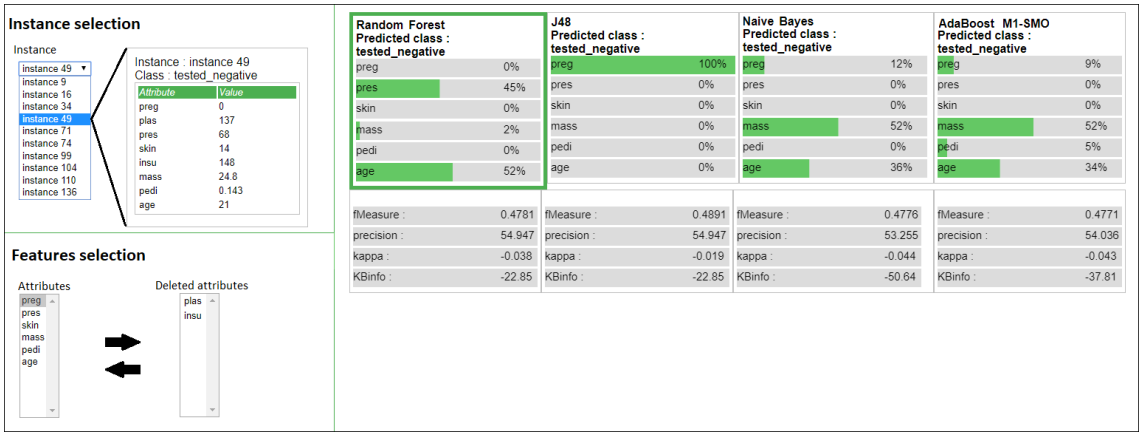


Figure 7.5: New prediction explanations once the attributes plasma and insulin have been removed.

of a feature and decide if the workflows are using them well or not.

7.4 Evaluation prototype

To test this framework further, it is necessary to develop a true prototype and assess its effectiveness through user reviews.

7.4.1 General architecture of the prototype

This prototype is realized using java, notably for the well-developed weka library, giving us a sane and practical basis. Weka has various objects for managing data and datasets, as well as training and storing data analysis models on a large array of situations and options. Moreover, this architecture is compatible with the Openml java interface. Openml's java library has been created with the base weka data structure in mind, along with the basic ARFF format[2] for storing and loading the different datasets. The man-machine interface of the prototype has been realized using GWT (Google Web Toolkit), a java-javascript interface, to create a web interface easily. We have an accessible version of the prototype on an apache server, hosted on a virtual machine in the IRIT laboratory. It can be found at `http://141.115.26.23/predinsight/` and has been dubbed "predinsight".

7.4.2 First evaluation : sandbox usage by a dentist

As a preliminary test of our prototype as a proof of concept, we worked with the INSERM laboratory, with which we frequently discussed. They gave us valuable feedback on our

[2]Weka definition at `https://www.cs.waikato.ac.nz/~ml/weka/arff.html`

prototype while we were developing it. Finally, we particularly worked with Paul Monsarrat, DDS, and Ph.D. of the STROMALab laboratory of the University of Toulouse. He and his team were interested in testing our prototype through their data collected when studying dental health evolution regarding everyday life habits. Thanks to this, we run the entirety of our whole system through real-life data. This example is illustrated through Figures 7.6 to 7.11, each figure corresponding to a step in the analysis.

- **Figure 7.6** First, the user uploads his dataset on Openml. This platform stores the user's experiment and compares it to other experiments of the Openml repository. Here, our dataset is a collection of patients' oral hygiene habits. The class is the presence of periodontitis, a classic oral infection caused by age and poor oral hygiene. Its values can be "absent", "gingivorrhagia" (bleeding of the mouth's gum, one of the first signs of periodontitis), and "provoked". The attributes of the dataset are the Frequency of visits to the dentist, the time elapsed since the last teeth scaling, the number of daily teeth brushing, the number of daily uses of dental floss, and the number of daily uses of mouthwash.

- **Figure 7.7** Once the dataset has been uploaded to Openml, it is automatically analyzed and its main characteristics are displayed for the user, allowing him to see his dataset with different visual representations and new statistics.

- **Figure 7.8** When the user is satisfied with his dataset and its characteristics, the user can click "recommend". The dataset is then compared to previous experiments present in Openml. Similar datasets are then found by the prototype, and promising workflows are displayed to the user. In our example, the user decides to select all the five recommended workflows, in order to see all his options.

- **Figure 7.9** The five workflows are then applied to the user's dataset. These workflows are realized through a 5-fold validation training process. We then generate for each instance the predictions of each model and thus the explanation of each prediction. As having to explore the whole dataset would be too much for a human, a set of 10 instances are selected and presented to the user. These prediction explanations allow the user to evaluate the different workflows. Here, he can see that the attribute "Frequency of visits to the dentist" poses a problem. indeed patients with periodontitis tend to go to the dentist to take care of their problem. Conversely, someone without any dental problems tends to visit less their dentist. This leads a lot of the

models to use this attribute in their prediction, but we can see that the fact that a person does not visit their dentist often is interpreted as a sign that they don't have periodontitis. This analysis is technically correct, but is not interesting for the user, as they want to study the causes of periodontitis, and visits to the dentists are finally more a consequence. Because of that, they decide to delete the attribute from their dataset. The different workflows are then run again on the model, without the attribute.

- **Figure 7.10** Once the models have been trained again on the dataset, the user can see their new prediction explanations and global performances. As an example, for the instance 136, they can see that three of the models use the frequency of mouthwashes and teeth scaling for their predictions, and very few use the teeth brushing frequency. With this information and the global performances of the models, the user can select his favorite model use as a predictive model for his future patients.

- **Figure 7.11** This model having been selected, the user can now enter his new patients as new instances of his dataset and predict their risks of having periodontitis. These predictions are still accompanied by their explanation. As in our example in the figure, the patient is predicted as not having any periodontitis. The explanation tells the user that the teeth brushing frequency, dental floss use, and the time since their last teeth scaling are signs in favor of the absence of periodontitis, while their daily use of mouthwash is lowering their chance of not having periodontitis, according to the model.

  Those results, depending on his intent, can give him insight when making decisions for his patients, but also help him explore new possible causes of periodontitis, by applying different hypotheses on the predictive model.

7.5 Second evaluation : A larger scale trial

It is now necessary to assess the interest of the public intended for our system through our prototype. This can be done via an experiment aiming to assess the effectiveness of our framework and the interest of our application of prediction explanation. This experiment is made with the assistance of the INSERM laboratory, with the participation of 20 students and 10 researchers in biology. These participants have a varying degree of mastery of data analysis, some of them have already done simple data analysis but never used its more complex tools, like machine learning. These participants are first given a demonstration of

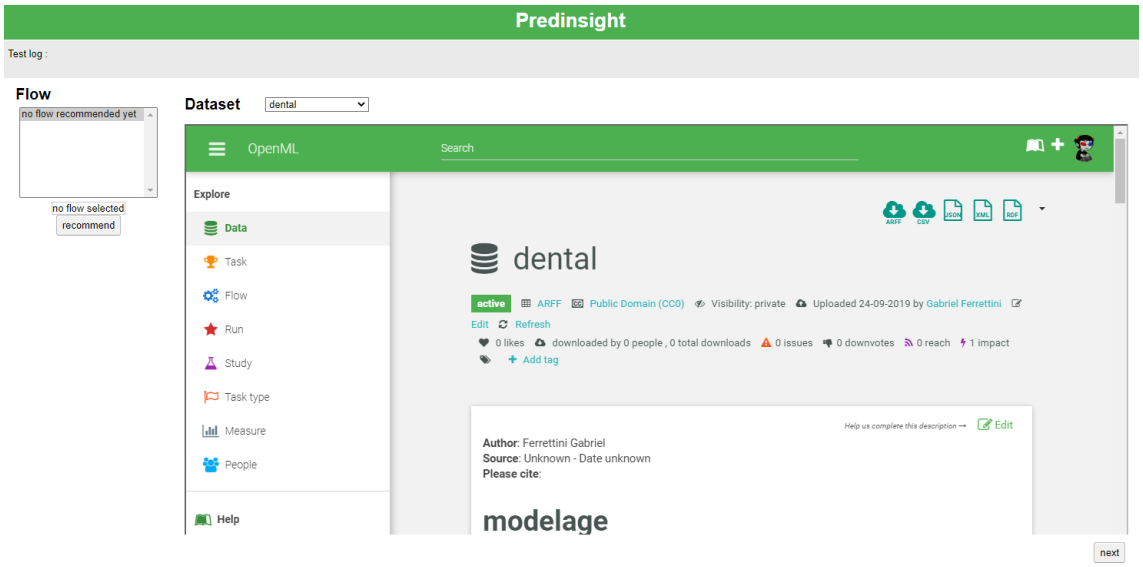


Figure 7.6: First screen of our prototype, prior to recommendation.

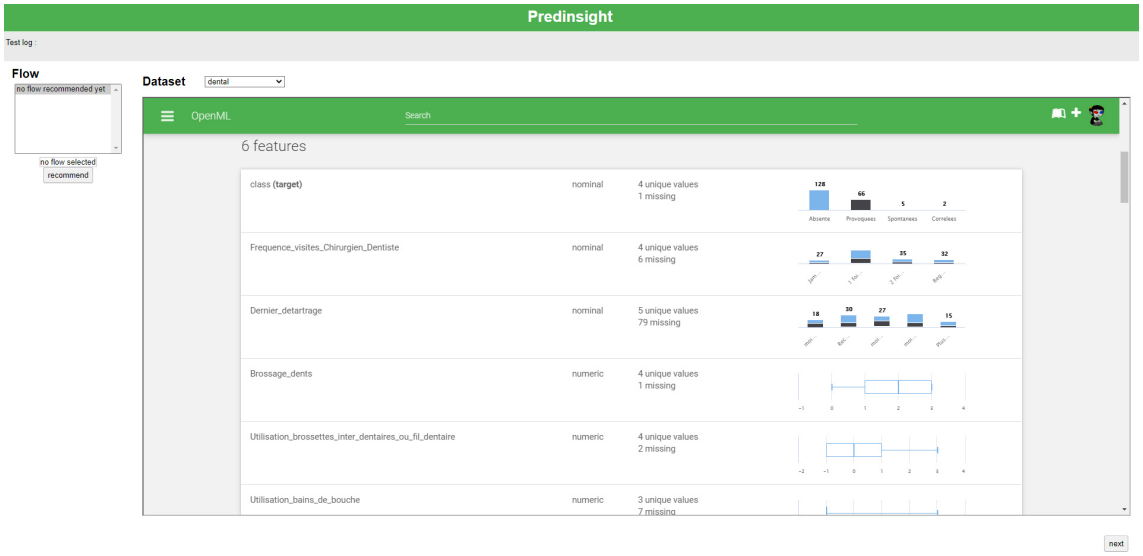


Figure 7.7: Openml allows us to access a useful visual representation of the dataset and its characteristics.

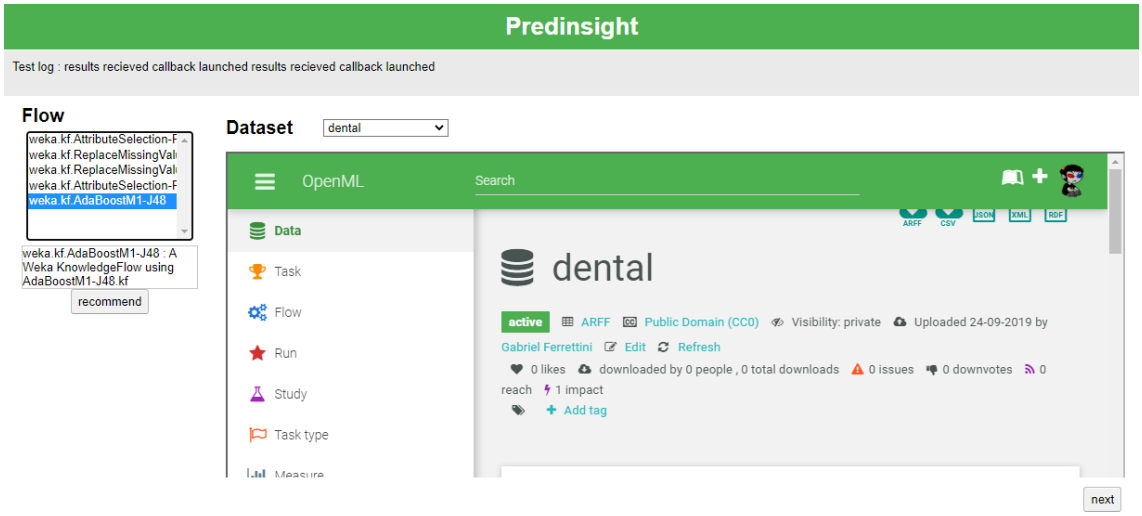


Figure 7.8: Once the workflows have been recommended, their description can be accessed, and they can be selected for the next step.

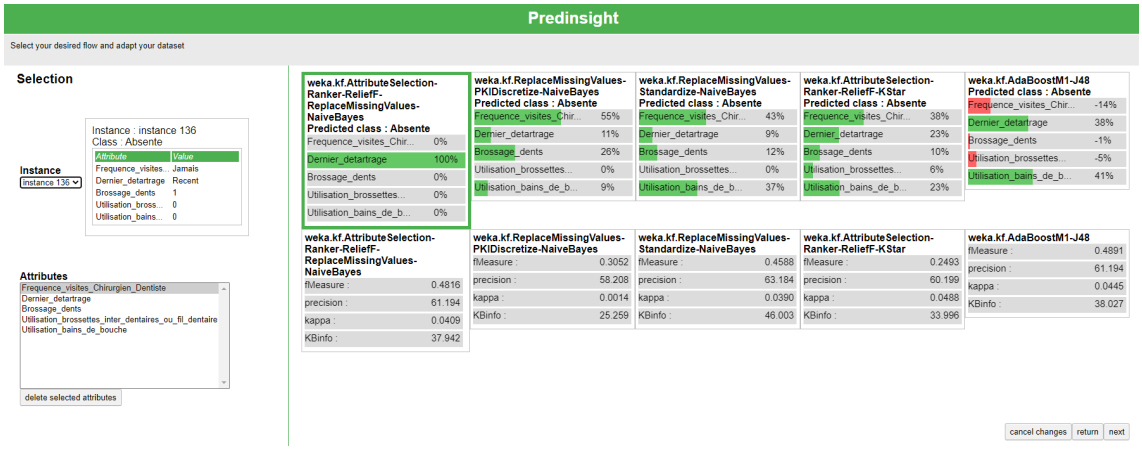


Figure 7.9: Second screen of our prototype, displaying the prediction explanation and global statistics for every models.

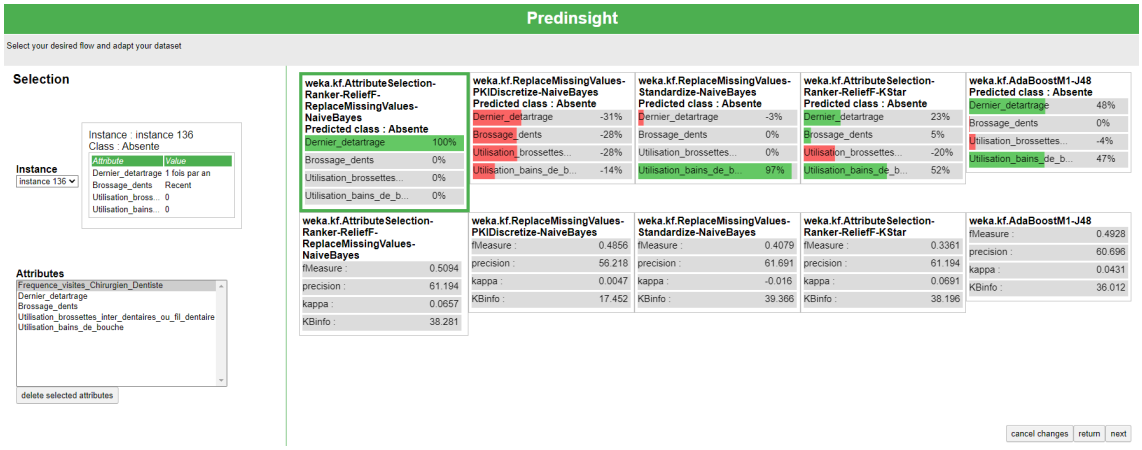


Figure 7.10: Once the attributes have been modified, we can see the evolution in the attributes importance in the prediction of each model.

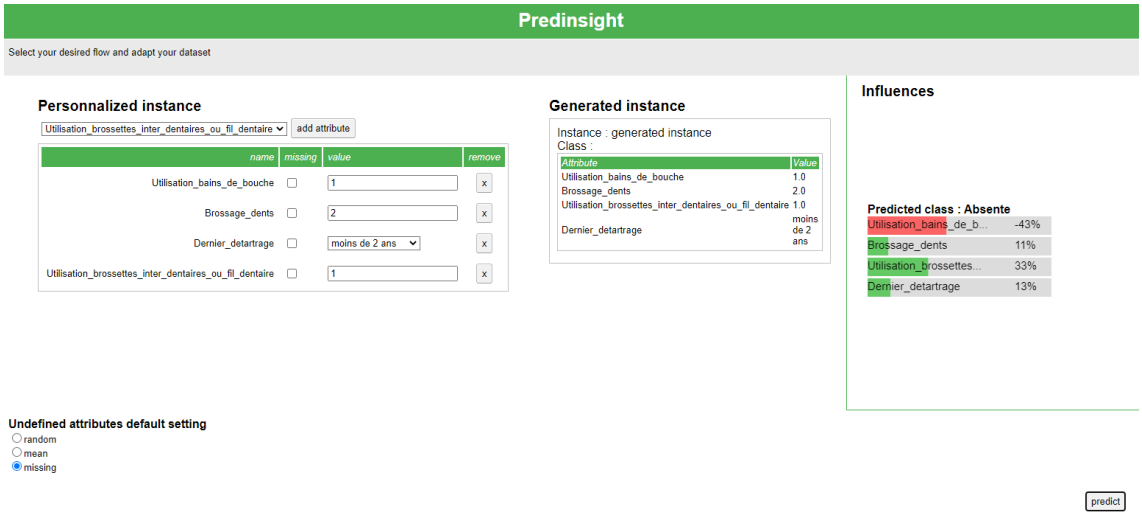


Figure 7.11: Third screen of our prototype, where the user can perform new predictions with the selected model.

the predinsight tool and its functionalities. They are then given a data analysis scenario described in annex, Section A.1. This scenario has been redacted with the help of Paul Monsarrat. The goal of this scenario is to give a context for the participants on which to use our prototype. It also guides them through the different steps of creating a workflow and gives them simple tasks to solve along the way.

7.5.1 Questions form and system evaluation

During the experimental scenario, a set of questions are asked during the experimentation in order to assess the state of the mind of the user about the prototype while using it. After the experiment, these users are then asked to fill a form that aims to assess their general impression of the prototype. It's question pertain to the confidence of the user in their decision, and ask them if they felt informed in their decisions or not. It also asks them about their previous background in machine learning and data analysis and their familiarity with the analyzed data.

Question form :

1. How familiar are you with data analysis?
    - Not at all

- A little
- Familiar
- Very familiar

2. How familiar are you with the analysed dataset?
   - Not at all
   - A few
   - Familiar
   - Very familiar
3. Did you understand the results produced by the application?
   - Not at all
   - A few
   - I understood the majority of it
   - I understood everything
4. Would you know how to use the results produced by the application?
   - Not at all
   - I'm not sure
   - Mostly
   - Completely
5. Concerning the automatically selected instance, how well did they allow you to explore the model?
   - Not at all
   - Not enough
   - Quite well
   - Very well
6. Which data analysis model did you select?

   I selected :
7. Why did you select it? (multiple answers possible)

- I chose it at random
- I know its name
- It was the fastest one
- I thought it was pertinent for this problem
- It's description seemed pertinent
- other :

8. Do you understand the produced prediction explanations?
   - Not at all
   - A few
   - Mostly
   - Completely
9. Did the prediction explanation help you choose among the recommended models?
   - Not at all
   - A little
   - A lot
   - Completely (I could not have chosen without it)
10. Did the prediction explanation help you understand the predictions of the model?
    - Not at all
    - A little
    - A lot
    - Completely
11. How confident are you in the results produced by the application?
    - Not at all
    - A little
    - Mostly confident
    - Very confident
12. Are you satisfied with the results produced by the application?

- Not at all
- Not very satisfied
- Satisfied
- Very satisfied

13. Would you have a use for such an application in your everyday job?

- Never/very rarely
- Sometimes
- Often
- Almost everyday

Along with those questions, we also evaluate the use of the system through the activity of the users. We monitor the results of the created machine learning predictive models and evaluate their performances to know if the processes brought a useful result to the users. Finally, we track their actions on the prototype to constitute a history of their actions, which gives us more information on their usage of the process.

Sadly, this experiment was scheduled to happen during the months of March or April. Consequently, the sanitary conditions of the SARS-CoV-2 pandemic prevented us to perform this experiment safely. Thus, it has not been realized as of the time of writing of this thesis.

7.6 Conclusion

During this chapter, we created a framework aiming to guide a user through the different steps of data analysis. By this framework, we combine the diverse elements developed through this thesis, to create a decision assistance system for creating a machine learning model. With it, we guide the user through the different steps of the choice and training of a model, by exploiting our prediction explanation system.

With those explanations, the user can spot eventual mistakes in the training of his model, choose between different possible workflows, and decide which attributes are pertinent in his data. Moreover, once the model is trained, the user has a sandbox environment at his disposal for experimenting with new data.

To verify these properties, we devised a prototype applying the framework to real case analysis. An experiment aiming to assess the effectiveness of this prototype has been developed in this chapter. Even if it could not have been brought to fruition, for the time being, we are eagerly waiting for its results.

Nonetheless, the prototype has received a lot of attention and positive reactions during its presentations as a demonstration during the PFIA and APSEM conventions of 2019. Moreover, a lot of the researchers working at the INSERM expressed a lot of interest in the framework. These are encouraging signs for this prototype, but they still need to be tested in a formal experiment before its effectiveness can be properly assessed.

This framework could be extended by developing its sandbox aspect. As an example, adding functionalities akin to those found in the What-if tool, combined with our prediction explanation system. This extended sandbox environment would let the user experiment with more complex hypotheses easily while having explanations at his disposal for an easier interpretation and comprehension.

References

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “"Why Should I Trust You?": Explaining the Predictions of Any Classifier”. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. event-place: San Francisco, California, USA. New York, NY, USA: ACM, 2016, pp. 1135–1144.

when life gives you lemons, call them golden oranges and sell them for double the price

---

Stanley Pines, Alex Hirsch

No amount of skill will protect you from the sheer luck of a chronic dumbass

---

polygonfighter

Chapter 8

Synthesis and perspective

During this Ph.D. thesis, our objective has been to **help a user at building his own data analysis model, independently of the expertise of the user in data analysis**. This problem is the center of numerous research and developments, as is proven by a large number of tools and solutions proposed in the literature. As we have seen in Part I, many of these solutions are more oriented toward facilitating the work of the data analysis expert than guiding non-expert users. Concurrently, the majority of solutions aiming to help non-expert on the data analysis process tend to eliminate human participation partially or totally. If this approach is effective in creating predictive models for a given dataset, these models do not always have the confidence of the user, as their functionalities remain unknown to him. Because of this problem, we aimed to put the human user back in the data analysis process, while guiding him toward the creation and training of the final model.

8.1 Contributions

First, we aimed to answer the issue of **efficiently recommending a data analysis workflow to a non-expert user**. In Chapter 5, we proposed a recommendation system as a basis of our work. This recommendation system does not aim to replace the user's decisions entirely, but rather to give him a possible solution without forcing his hand toward a particular choice. It is based on a dissimilarity measure, which we use to compare the user's dataset with other datasets from a repository of previous experiments. As we aimed to keep the user involved in the process of data analysis, our proposal uses the user's preferences in terms of performance and evaluation metrics. This makes the recommender system a context-based collaborative filtering recommendation system, which involves the

user in his recommendation process.

This recommendation system shows promising results, but do not allow the user to understand what is being recommended to him. To address our issue of **help a non-expert understand and analyze the results produced by a data analysis process**, we focused on a solution for explaining a predictive model and bypass their black box aspect.

Thus, during chapter 6 we then proposed a solution for explaining the predictions of a machine learning model. As it is not possible to access directly the inner working of a machine learning model, we decided to investigate the *results* produced by a model. To do so, we based our work on the principles of additives explanations, which explains the models by quantifying the importance of each attribute of the dataset when the model is making a prediction. We identified that those techniques tend to have flaws: either they are specifically designed for a particular type of model, or producing accurate explanation comes at the cost of an important computational load. Thus, we proposed a prediction explanation system, which functions by identifying the most important information to be considered when calculating the explanation. For this, we created multiples methods for determining which information would be calculated. We then use it to focus our calculations and strike a balance between precision and computation time. These different methods have been tested against each other on a large collection of datasets. The results have put in light the interests of diverse methods, depending on the context in which the method is applied. Notably, the more precise methods tend to have their computation time grow rapidly with the number of attributes of the studied dataset, while the less precise are faster to compute, without losing too much information. Thus, a satisfactory approach to explain the recommended models can be selected depending on the size of the user's dataset.

It then became necessary to study how to apply our proposals to a real-life user to answer our main issue: **How can we help a user who is not an expert in data analysis at building his own data analysis model?**.

Our previous propositions have been combined in a new framework in chapter 7. By applying prediction explanation in the context of model selection and attributes selection, it becomes possible for a user to act upon important decisions for the machine learning process, without requiring expertise in the domain.

In terms of model selection, the user can assess the importance of each attribute in the predictions of each model. This allows him to access the "thought process" of each model,

and evaluate them on the field of his expertise. With that approach, the user can rely on his knowledge of his field and then assess the importance of the dataset's attributes.

Regarding attributes selection, through prediction explanation, the user can assess the usefulness of the different attributes of his dataset. It can lead him to make different decisions regarding the attributes, notably the user can decide that an attribute should not have been added to the dataset in the first place. Another main application of this approach is when a user considers an attribute as useless after the fact, or that the models focus too much on it. Thus, prediction explanation enables a non-expert user to perform attribute and model selection.

8.2 Perspectives

We have created a novel framework for assistance in data analysis, by using our different solutions and processes to help a non expert user in an original way. With this framework, we have answered a number of drawbacks to the use of machine learning for expert users. Through these processes, we have created a process facilitating the creation of predictive models, without removing the user entirely from it. By designing our prediction explanation system, we answer the problem of lack of comprehension from the user in a faster and more comprehensive way than other methods from the state of the art. By using it in a novel way, we avoid the confidence problem usually brought by black-box machine learning models.

Although, some enhancements can be brought to this framework. In a short term perspectives, our framework still needs to be tested in a real-world situation in order to really assess its usefulness.

Our recommendation process currently takes into account the preferences of the user, and the results produced by it are encouraging, but this user profile might be extended. As an example, our recommender system might take into account the domain of expertise of the user, his current line of activity, or any other information relevant to his current task.

Regarding prediction explanations, our method is generally faster than the others while keeping a high precision in its explanations. However, datasets with more than a dozen attributes might cause an explosion in the calculation cost. Thus, other solutions might be explored for a larger dataset, such as selecting a set of the ten most important attributes in a global sense and only running a single instance explanation for those attributes. By limiting the explanation to ten attributes, we avoid the calculatory explosion caused by

a too large number of attributes. Moreover, by lowering the number of attributes in the explanation, we also avoid overwhelming the user with too much information.

Our prediction explanation framework may also be extending by using the user's actions as feedback for improving the recommendations further. By learning from the user's choices, the recommendation system might be modified to favor the models most useful to him, as an example.

Moreover, when the user is performing model selection, our framework selects a set of instances to recommend to the user to provide him with interesting data points for the exploration of the model. This method of selecting instances might be explored further, particularly depending on what we want to present to the user. We might want to show him the particularities of his dataset. Another possibility is to show him the particularities of the model. We might also want to present to him the most problematic data points for the models. All these approaches are valid and can bring important information to the user, thus it would be interesting to consider how to select those instances, and what information can be retrieved from the selection.

From a longer-term perspective, we could expand the sandbox aspect of our framework by allowing the user to test hypotheses on his data or his model. The user could formulate hypotheses about his data, and see the effects of those hypotheses on the prediction explanations and results of the model. As an example, the what-if tool proposes interesting solutions for allowing a user to test the changes provoked by modifying the data or the model, as changing the mean of an attribute or its standard deviation. Combining those ideas to our framework could produce an even greater sandbox environment.

Finally, as we want to involve the user as much as possible in the data analysis process, it might be interesting to extend our propositions to the whole machine learning workflow. Our framework could allow the user to build a modular workflow by himself rather than being proposed a set of single-block workflows to choose from. By combining recommendations and explanations, the user might be guided through a more precise construction of his machine learning model. As an example, prediction explanation could be used to create a modular workflow building framework, based on systems similar to orange, using the explanations to make the different steps and modifications more approachable for a non-expert user.

Complete bibliography

Bibliography

[Abd18] Salisu Mamman Abdulrahman et al. “Speeding up algorithm selection using average ranking and active testing by introducing runtime”. In: *Machine learning* 107.1 (2018), pp. 79–108.

[Adk13] Dustin Adkison. *IBM Cognos Business Intelligence*. Packt Publishing Ltd, 2013.

[Ado05] Gediminas Adomavicius et al. “Incorporating contextual information in recommender systems using a multidimensional approach”. In: *ACM Transactions on Information Systems (TOIS)* 23.1 (2005), pp. 103–145.

[Ali15] Julien Aligon et al. “A collaborative filtering approach for recommending {OLAP} sessions”. In: *Decision Support Systems* 69 (2015), pp. 20–30.

[Alt10] André Altmann et al. “Permutation importance: a corrected feature importance measure”. In: *Bioinformatics* 26.10 (2010), pp. 1340–1347.

[AT08] Gediminas Adomavicius and Alexander Tuzhilin. “Context-aware Recommender Systems”. In: *Proceedings of the 2008 ACM Conference on Recommender Systems*. RecSys ’08. event-place: Lausanne, Switzerland. New York, NY, USA: ACM, 2008, pp. 335–336.

[AT11] Gediminas Adomavicius and Alexander Tuzhilin. “Context-aware recommender systems”. In: *Recommender systems handbook*. Springer, 2011, pp. 217–253.

[BBM07] D Bourgeois, P Bouchard, and C Mattout. “Epidemiology of periodontal status in dentate adults in France, 2002–2003”. In: *Journal of periodontal research* 42.3 (2007), pp. 219–227.

[Ber06] Pavel Berkhin. “A survey of clustering data mining techniques”. In: *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[Ber07] Michael R. Berthold et al. “KNIME: The Konstanz Information Miner”. In: *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. ISSN: 1431-8814. Springer, 2007.

[BHM04] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. “Query Recommendation Using Query Logs in Search Engines”. In: *Proceedings of the 2004 International Conference on Current Trends in Database Technology*. EDBT’04. event-place: Heraklion, Greece. Berlin, Heidelberg: Springer-Verlag, 2004, pp. 588–596. DOI: 10.1007/978-3-540-30192-9_58.

[Bra08] Pavel Brazdil et al. *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

[BS85] S.W. Bennett and A.C. Scott. "The Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, chap. 19 - Specialized Explanations for Dosage Selection". In: *Addison-Wesley Publishing Company* (1985), pp. 363–370.

[BSD03] Pavel B Brazdil, Carlos Soares, and Joaquim Pinto Da Costa. "Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results". In: *Machine Learning* 50.3 (2003), pp. 251–277.

[CMB18] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. "Visualizing the feature importance for black box models". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2018, pp. 655–670.

[Coh68] Jacob Cohen. "Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit." In: *Psychological bulletin* 70.4 (1968). Publisher: American Psychological Association, p. 213.

[CP05] Li Chen and Pearl Pu. "Trust Building in Recommender Agents". In: *in 1st International Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces (WPRSIUI05*. 2005, pp. 135–145.

[Cra08] Henriette S. M. Cramer et al. "The effects of transparency on trust in and acceptance of a content-based art recommender". In: *User Model. User-Adapt. Interact.* 18.5 (2008), pp. 455–496. DOI: 10.1007/s11257-008-9051-3.

[Dem06] Janez Demšar. "Statistical comparisons of classifiers over multiple data sets". In: *Journal of Machine learning research* 7.Jan (2006), pp. 1–30.

[Dem13] Janez Demšar et al. "Orange: Data Mining Toolbox in Python". In: *Journal of Machine Learning Research* 14 (2013), pp. 2349–2353.

[DK17] Finale Doshi-Velez and Been Kim. "Towards a rigorous science of interpretable machine learning". In: *arXiv preprint arXiv:1702.08608* (2017).

[DSZ16] A. Datta, S. Sen, and Y. Zick. "Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems". In: *2016 IEEE Symposium on Security and Privacy (SP)*. ISSN: 2375-1207. May 2016, pp. 598–617.

[ElS19] Radwa ElShawi et al. "ILIME: Local and Global Interpretable Model-Agnostic Explainer of Black-Box Decision". In: *European Conference on Advances in Databases and Information Systems*. Springer. 2019, pp. 53–68.

[Feu15] Matthias Feurer et al. "Efficient and robust automated machine learning". In: *Advances in neural information processing systems*. 2015, pp. 2962–2970.

[Feu19] Matthias Feurer et al. "Auto-sklearn: efficient and robust automated machine learning". In: *Automated Machine Learning*. Springer, Cham, 2019, pp. 113–134.

[Ges17] Felix Gessert et al. “NoSQL database systems: a survey and decision guidance”. In: *Computer Science-Research and Development* 32.3-4 (2017), pp. 353–365.

[GG99] Michael Goebel and Le Gruenwald. “A survey of data mining and knowledge discovery software tools”. In: *ACM SIGKDD explorations newsletter* 1.1 (1999), pp. 20–33.

[Gia09] Arnaud Giacometti et al. “Query recommendations for OLAP discovery driven analysis”. In: *Proceedings of the ACM twelfth international workshop on Data warehousing and OLAP*. 2009, pp. 81–88.

[GT20] Filippo Graziani and Georgios Tsakos. “Patient-based outcomes and quality of life”. In: *Periodontology 2000* 83.1 (2020), pp. 277–294.

[GVB04] Christophe Giraud-Carrier, Ricardo Vilalta, and Pavel Brazdil. “Introduction to the special issue on meta-learning”. In: *Machine learning* 54.3 (2004), pp. 187–193.

[Haa11] Peter J Haas et al. “Data is dead... without what-if models”. In: *Proceedings of the VLDB Endowment* 4.12 (2011), pp. 1486–1489.

[Hal09] Mark Hall et al. “The WEKA data mining software: an update”. In: *ACM SIGKDD explorations newsletter* 11.1 (2009). Publisher: ACM, pp. 10–18.

[HC10] Jianming He and Wesley W. Chu. “A Social Network-Based Recommender System (SNRS)”. In: *Data Mining for Social Network Data*. Ed. by Nasrullah Memon et al. Boston, MA: Springer US, 2010, pp. 47–74. DOI: 10.1007/978-1-4419-6287-4_4.

[Hen14] Andreas Henelius et al. “A peek into the black box : exploring classifiers by randomization”. In: *Data mining and knowledge discovery* 28.5-6 (2014). Publisher: Stockholms universitet, Institutionen för data- och systemvetenskap, pp. 1503–1529.

[HK13] Markus Hofmann and Ralf Klinkenberg. *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Chapman & Hall/CRC, 2013.

[HPU17] Andreas Henelius, Kai Puolamäki, and Antti Ukkonen. “Interpreting classifiers through attribute interactions in datasets”. In: *arXiv preprint arXiv:1707.07576* (2017).

[Ido18] Carlie Idoine et al. “Magic Quadrant for data science and machine-learning platforms”. In: *Gartner, Inc* (2018).

[IY17] U. A. Piumi Ishanka and Takashi Yukawa. “The Prefiltering Techniques in Emotion Based Place Recommendation Derived by User Reviews”. In: *Applied Computational Intelligence and Soft Computing* vol. 2017 (2017), 10 pages. DOI: doi:10.1155/2017/5680398.

[KB91] Igor Kononenko and Ivan Bratko. “Information-Based Evaluation Criterion for Classifier’s Performance”. In: *Machine Learning* 6.1 (Jan. 1991), pp. 67–80.

[KR92] Kenji Kira and Larry A Rendell. “A practical approach to feature selection”. In: *Machine Learning Proceedings 1992*. Elsevier, 1992, pp. 249–256.

[Kri16] Sanjay Krishnan et al. “Towards reliable interactive data cleaning: A user survey and recommendations”. In: *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. 2016, pp. 1–5.

[KSA08] Albert Kingman, Cristiano Susin, and Jasim M Albandar. “Effect of partial recording protocols on severity estimates of periodontal disease”. In: *Journal of clinical periodontology* 35.8 (2008), pp. 659–667.

[LBG15] Christiane Lemke, Marcin Budka, and Bogdan Gabrys. “Metalearning: a survey of trends and technologies”. In: *Artificial intelligence review* 44.1 (2015), pp. 117–130.

[LBV12] Rui Leite, Pavel Brazdil, and Joaquin Vanschoren. “Selecting classification algorithms with active testing”. In: *International workshop on machine learning and data mining in pattern recognition*. Springer. 2012, pp. 117–131.

[Lin10] Shili Lin. “Rank aggregation methods”. In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.5 (2010), pp. 555–570.

[LL17a] Scott M Lundberg and Su-In Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4765–4774.

[LL17b] Scott M Lundberg and Su-In Lee. “Consistent feature attribution for tree ensembles”. In: *arXiv preprint arXiv:1706.06060* (2017).

[Mak19] Sara Makki. “An Efficient Classification Model for Analyzing Skewed Data to Detect Frauds in the Financial Sector”. PhD thesis. Université de Lyon; Université libanaise, 2019.

[MAT10] MATLAB. *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.

[Mon16] Paul Monsarrat et al. “Clinical research activity in periodontal medicine: a systematic mapping of trial registers”. In: *Journal of clinical periodontology* 43.5 (2016), pp. 390–400.

[NHK14] Phong Nguyen, Melanie Hilario, and Alexandros Kalousis. “Using meta-mining to support data mining workflow planning and optimization”. In: *Journal of Artificial Intelligence Research* (2014), pp. 605–644.

[NMY14] Saggi Neuman, Moty Michaely, and MOR Yaniv. *System and method for management of big data sets*. US Patent App. 14/052,785. June 2014.

[PBG00] Bernhard Pfahringer, Hilan Bensusan, and Christophe G Giraud-Carrier. “Meta-Learning by Landmarking Various Learning Algorithms.” In: *ICML*. 2000, pp. 743–750.

[Qui86] J.R. Quinlan. “Induction of Decision Trees”. In: *Machine Learning* 1.1 (Mar. 1986), pp. 81–106.

[Ray18] William Raynaut. “Perspectives de Méta-Analyse pour un Environnement d’aide à la Simulation et Prédiction”. francais. Thèse de doctorat. Toulouse, France: Université de Toulouse, Université Toulouse III-Paul Sabatier, Jan. 2018.

[Rij15] Jan N van Rijn et al. “Fast algorithm selection using learning curves”. In: *International symposium on intelligent data analysis*. Springer. 2015, pp. 298–309.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “"Why Should I Trust You?": Explaining the Predictions of Any Classifier”. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’16. event-place: San Francisco, California, USA. New York, NY, USA: ACM, 2016, pp. 1135–1144.

[RSV16] William Raynaut, Chantal Soule-Dupuy, and Nathalie Valles-Parlangeau. “Meta-Mining Evaluation Framework : A large scale proof of concept on Meta-Learning”. In: *29th Australasian Joint Conference on Artificial Intelligence*. (Classée B par core.edu.au). Springer, Dec. 5, 2016, pp. 215–228.

[RSV17] William Raynaut, Chantal Soulé-Dupuy, and Nathalie Vallès-Parlangeau. “Dissimilarités entre jeux de données”. In: *Ingénierie des Systèmes d Inf.* 22.3 (2017), pp. 35–63. DOI: 10.3166/isi.22.3.35-63.

[SGK17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning Important Features Through Propagating Activation Differences”. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML’17. event-place: Sydney, NSW, Australia. 2017, pp. 3145–3153.

[Sha13] Guy Shani et al. “Investigating confidence displays for top-N recommendations”. In: *Journal of the American Society for Information Science and Technology* 64.12 (2013), pp. 2548–2563.

[Sha16] Mahmood Sharif et al. “Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition”. In: *Proceedings of the 2016 acm sigsac conference on computer and communications security*. 2016, pp. 1528–1540.

[Sha53] L. S. Shapley. “A value for n-person games”. In: *Contributions to the Theory of Games* 28 (1953). Publisher: Princeton University Press, pp. 307–317.

[ŠK08a] Erik Štrumbelj and Igor Kononenko. “Towards a Model Independent Method for Explaining Classification for Individual Instances”. In: *Data Warehousing and Knowledge Discovery*. Ed. by Il-Yeol Song, Johann Eder, and Tho Manh Nguyen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 273–282.

[ŠK08b] Erik Štrumbelj and Igor Kononenko. “Towards a model independent method for explaining classification for individual instances”. In: *International Conference on Data Warehousing and Knowledge Discovery*. Springer. 2008, pp. 273–282.

[SK10] Erik Strumbelj and Igor Kononenko. “An Efficient Explanation of Individual Classifications Using Game Theory”. In: *J. Mach. Learn. Res.* 11 (Mar. 2010). Publisher: JMLR.org, pp. 1–18.

[SO14] Rachel Schutt and Cathy O'Neil. *Doing data science: Straight talk from the frontline*. O'Reilly, 2014.

[Spa08] Concetto Spampinato et al. "Detecting, Tracking and Counting Fish in Low Quality Unconstrained Underwater Videos." In: *VISAPP (2)* 2008.514-519 (2008), p. 1.

[Sto74] M. Stone. "Cross-Validatory Choice and Assessment of Statistical Predictions". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 36.2 (1974), pp. 111–147. DOI: 10.2307/2984809.

[TM15a] Nava Tintarev and Judith Masthoff. "Explaining Recommendations: Design and Evaluation". In: *Recommender Systems Handbook*. Ed. by Francesco Ricci, Lior Rokach, and Bracha Shapira. Boston, MA: Springer US, 2015, pp. 353–382. DOI: 10.1007/978-1-4899-7637-6_10.

[TM15b] Nava Tintarev and Judith Masthoff. "Explaining recommendations: Design and evaluation". In: *Recommender systems handbook*. Springer, 2015, pp. 353–382.

[Ton17] Maurizio S Tonetti et al. "Impact of the global burden of periodontal diseases on health, nutrition and wellbeing of mankind: A call for global action". In: *Journal of clinical periodontology* 44.5 (2017), pp. 456–462.

[Van13] Joaquin Vanschoren et al. "OpenML: Networked Science in Machine Learning". In: *SIGKDD Explorations* 15.2 (2013). Place: New York, NY, USA Publisher: ACM, pp. 49–60. DOI: 10.1145/2641190.2641198.

[Van18] Joaquin Vanschoren. "Meta-Learning: A Survey". In: *CoRR* abs/1810.03548 (2018).

[Wan17] Qi Wang et al. "A context-aware researcher recommendation system for university-industry collaboration on R&D projects". In: *Decision Support Systems* 103.Supplement C (2017), pp. 46–57.

[Wex19] James Wexler et al. "The what-if tool: Interactive probing of machine learning models". In: *IEEE transactions on visualization and computer graphics* 26.1 (2019), pp. 56–65.

[WMR17] Sandra Wachter, Brent Mittelstadt, and Chris Russell. "Counterfactual explanations without opening the black box: Automated decisions and the GDPR". In: *Harv. JL & Tech.* 31 (2017), p. 841.

[Yan18] Chengrun Yang et al. "Oboe: Collaborative filtering for automl initialization". In: *arXiv preprint arXiv:1808.03233* (2018).

[ZMB15] Yong Zheng, Bamshad Mobasher, and Robin Burke. "Similarity-Based Context-Aware Recommendation". In: *Web Information Systems Engineering – WISE 2015: 16th International Conference, Miami, FL, USA, November 1-3, 2015, Proceedings, Part I*. Ed. by Jianyong Wang et al. Cham: Springer International Publishing, 2015, pp. 431–447.

Appendix A

A.1 Experimental scenario

A.1.1 Biomedical context : Studying periodontitis

The oral cavity is a proof of concept that can be easily generalized to the whole body. There are indeed strong bilateral relationships between oral health and general health ([Mon16]). Periodontal health (i.e. the supporting tissues of the teeth) is, in particular, an excellent reflection of the environmental footprint. In fact, the gingiva covers and protects against attack the alveolar-dental anchoring, whose integrity allows the maintenance of the tooth on the arch, the chewing function, and aesthetics. The multiple stresses that the oral cavity undergoes during life are therefore likely to form the basis of periodontal pathologies, in particular periodontitis.

Periodontitis is a chronic inflammatory disease of the supporting tissues of the teeth, which results from a breakdown in tissue homeostasis caused by a progressively deleterious ecosystem and an incompetent host's immune system. The disease is characterized by chronic gingival inflammation associated with the formation of lesions that combine gingival retraction and resorption of peri-radicular tissues, causing progressive mobility or even loss of the tooth (Figure A.1). The diagnosis is based on a set of clinical and radiological parameters including the degree of gingival inflammation and the level of alveolar bone around the teeth.

The prevalence of periodontitis is increasing and increasing with age. This chronic inflammatory condition is one of the most common in the world. In France, as in other developed countries, half of the population is affected by periodontitis in the 35-65 age group ([BBM07; KSA08]).

Without care, periodontitis leads to dental mobility then edentulousness, with medical consequences and their co-morbidities (such as alteration of nutrition or even digestive

Figure A.1: Periodontitis in a 67-year-old subject. The white and black arrows respectively indicate the inflammatory sites and the retracted and purulent areas, typical of this pathology (Toulouse University Hospital).

pathologies), psychological and social, profoundly altering the quality of life ([GT20]). Periodontitis, therefore, contributes to worsening direct and indirect health costs ([Ton17]). On the other hand, it is established that periodontitis can have an impact on systemic health with, for example, a direct impact on diabetes or cardiovascular pathologies. The presence of periodontitis is believed to be associated with more than 57 general pathologies ([Mon16]).

Periodontitis is, therefore, a major public health problem which will continue to increase with the increase in life expectancy.

If the diagnosis of established periodontitis is accessible by current clinical and radiological means of investigation, the early detection of alterations in the periodontium and the identification of populations at risk are necessary. The following scenario focuses on developing a predictive model to predict periodontium alterations based on medical and socio-demographic data, disregarding clinical data. Otherwise reformulated, is it possible to predict a periodontal state of health from simple questions, either self-reported by the patient, or that can be asked by telephone by a health professional.

A.1.2 Part 1: attributes selection and predictor variables identification

In the remainder of the experiment, the term “attribute” designates the variables used to develop the predictive model and the term “instance” the individuals for whom we have the attributes, that is to say, the individuals who answered the questions. The "predicted class" corresponds to the periodontal state of health that should be predicted by the algorithm (healthy, gingivitis, or periodontitis).

Step 1: choice of algorithms

The tool allows the use of 20 machine learning algorithms. Each algorithm has its specificities, strengths, and weaknesses. The tool first recommends a set of 5 algorithms relevant to the present problem. You can then explore each algorithm's predictions and view the explanations provided. It is recommended to select the "Random Forest" and the "Naive Bayes" algorithms as they are algorithms which tend to have a behavior simple to understand.

After a calculation time, the tool displays the performance of each model. Usually, precision, recall, and F1 score are found as performance metrics for machine learning algorithms. Precision is the ratio between the number of individuals correctly assigned to a class and the total number of individuals assigned to that class (in statistics, this is the positive predictive value). The recall is the ratio between the number of individuals correctly assigned to a class and the total number of individuals belonging to this class (in statistics, this is sensitivity). The F1 score constitutes a value combining both the precision and the recall by taking their harmonic mean.

Step 2: Model explanation

Along with these general performance indicators, a set of 10 instances from the dataset is selected for you to explore. First, restore the full set of attributes for this step.

The tool allows you to understand which attributes were decisive (i.e. their weight) in the decision making of the algorithm for each individual in these instances. The presence of a green bar in the diagram means that the presence of this attribute value increased the probability of voting for the predicted class and the presence of a red bar in the diagram means that the presence of this value of attribute decreased the likelihood of voting for the predicted class. The associated value is the relative weight that this attribute has over all of the other attributes in predicting the class.

Example: in a model wishing to predict “absence of periodontal disease”, a value of the attribute “Age" at "43” of -40% means that for an individual aged 43 with this set of

attributes, the algorithm considered that this value lowered the chances that he did not have periodontitis. This attribute alone accounts for 40% of the model decision. Conversely, eating fruit once a day increases the likelihood of having a healthy periodontium according to the algorithm, with a weight of 10% of the model's decision.

Observe which variables come up most often in the explanation of the model when the model predicts the class correctly. Then observe the badly predicted instances (individuals). Interpret the variables that have an important weight in these bad decisions. What are the reasons you can evoke?

Step 3: Attributes selection

Make an initial selection of attributes by first keeping only the attributes that can be read over the phone and/or can be entered by the patient himself without the intervention of a healthcare professional. Observe and record the results of the metrics for each dataset, along with the models' explanations.

Then experience the impact of removing attributes from the models. For example, try to remove the attributes "sex" or "body mass index BMI" from the model. How do the metrics change? What is the impact on the model's explanation? What can you conclude from this?

Keep experimenting, trying to create the minimum model, that is, having the best balance between a small number of attributes and a good prediction rate with consistent reasoning of the model.

A.1.3 Part 2: Prediction on mockup patients

Now that you have developed different models for your dataset, select the one which suits you the most, per its performance metrics and its internal reasoning, reflected by the model's prediction explanation.

Now imagine that you are on the phone with different patients. You have never seen them, they are making an appointment for a first consultation. In order to better help you plan these first consultations, you ask the limited number of questions (i.e. attributes) that you need, given the optimized algorithm developed in the previous part.

A set of randomized patient is given to you to reflect these conversations.

Now enter the patients' attributes in your newly selected model, and observe the predictions made for each patient:

Are you confident in the predictions made by the model? What elements brought you to this conclusion?

Is your patient at risk for periodontal disease? What attributes allowed the model to make its decision?