Causal feature selection for predicting interventions

Auteur : Van Buggenhout, François
Promoteur(s) : Geurts, Pierre
Faculté : Faculté des Sciences appliquées
Diplôme : Master en ingénieur civil électricien, à finalité approfondie
Année académique : 2015-2016
URI/URL : http://hdl.handle.net/2268.2/1370

University of Liege - Faculty of Applied Sciences

Master Thesis
Causal feature selection for predicting interventions

Master thesis realized with the aim of obtaining the degree of Master in Electrical Engineering by **François Van Buggenhout**

Author:
François Van Buggenhout

Academic Supervisor:
Pierre Geurts

Academic year 2015-2016

Causal feature selection for predicting interventions

Van Buggenhout François

2^{nd} year of the degree of Master in Electrical Engineering

Academic year 2015-2016

Abstract

With the advent of high throughput experiments, researchers are more and more often confronted to really big datasets especially in biology. The number of variables can reach several thousand and to deal with this increasing number of features, researchers are using feature selection techniques to reduce the size of the datasets and so reduce the costs linked to the experimentation.

In machine learning, the objective is to learn a model on a dataset in order to predict the target variable of a second one drawn from the same distribution. Sometimes, the distribution is not the same between the two databases. For example, in biology, an experiment on the genes can be done on population from various places on earth. Since the genomes between different population present some variations, it is important to take them into account before selecting genes to predict the result of the experiment. These variations can also come from external agents like drugs which manipulate some features.

In this thesis, we focused on data generated from causal graph because it is representative of the phenomena met in biology like gene network. We were interested in a particular situation where the target is provided for an unmanipulated dataset but the objective is to predict the target of a manipulated database.

We developed a three stages process to find an optimal subset of features involving causal feature selection, filtering and two algorithms developed in the framework of this thesis.

The first step was to find a small set of features made of both highly correlated manipulated and unmanipulated variables. The two following steps focused on retrieving the manipulations and removing the bad features.

The whole process was able to significantly increase the prediction efficiency compared to classical feature selection techniques.

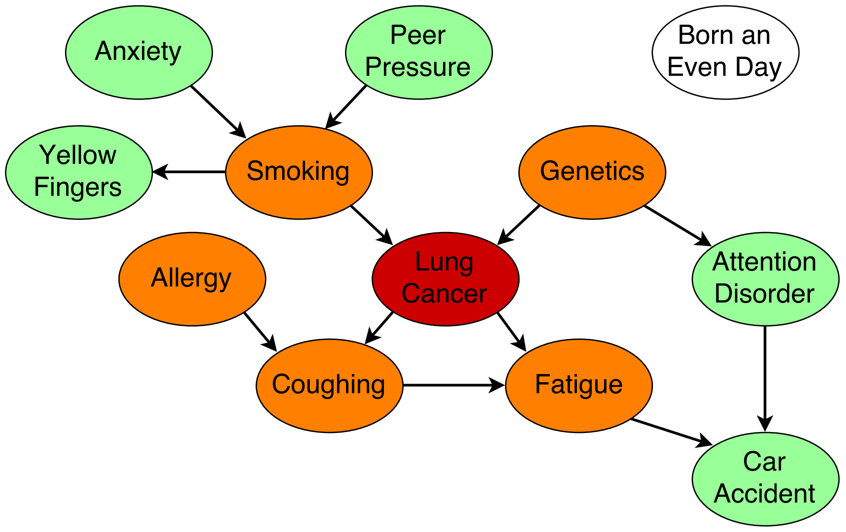


Figure 1: Graph of the LUCAS dataset showing the edges between the different features when there is no manipulation. The red, orange, green and white colors respectively represent the target, the Markov Boundary, redundant features and independent features.

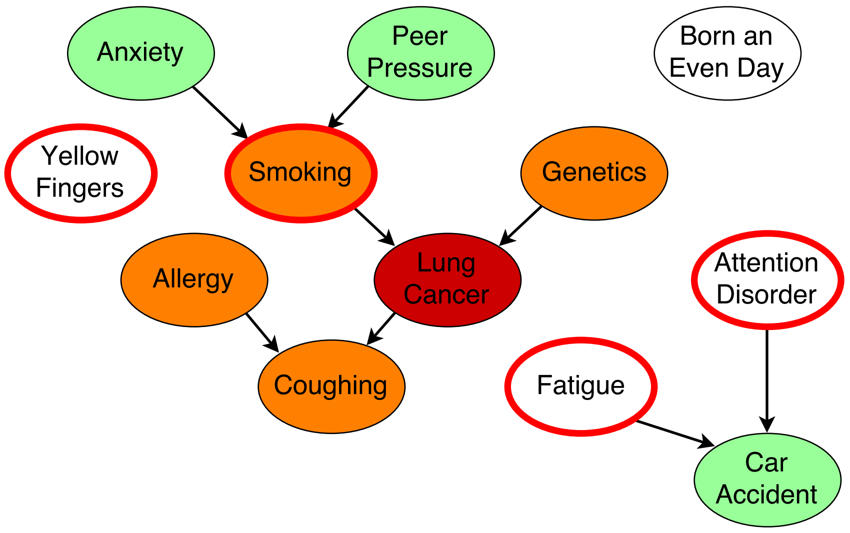


Figure 2: Graph of the LUCAS dataset showing the edges between the different features when there are some manipulations. The red, orange, green and white colors respectively represent the target, the Markov Boundary, redundant features and independent features. The features circled by a red line represent the manipulated features.

Acknowledgment

First I would like to thank my supervisor Professor Pierre Geurts for his guidance, innovative ideas and his availability.

I particularly want to express my gratitude to Vân Anh Huynh-Thu for her precious help through the different stages of this thesis. Many thanks for her availability and technical support.

I also want to thank all the participants to the LAB meeting for their constructive discussions.

Finally, I thank all those who have contributed to the achievement of this work, in one way or another, and who are not mentioned here.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since Newton, we know that any action causes a reaction but when you observe an experiment it is always quite hard to know which action generates which reaction. The reflection leading to the attribution of causes to effects is a part of our daily life involving analysis of a situation and decision making. Sometimes, it is not necessary to understand the whole process to be able to predict the final result of an experimentation even if some parameters are manipulated. Predicting effects of a particular experiment is important (for investor before modifying a policy) or vital (for patients) especially when the experiment is expensive or unethical (one can not force people to take a drug to see if it works to cure a specific disease).

1.1 Definition of the problem

In machine learning, feature selection is one of the most important step to reach good predictive efficiency. Moreover, some features may be expensive to sample and it is important to limit the number of features to avoid useless cost. Recently, people got interested by not only selecting features based on their information for prediction but they also want to know how these features interact with the target. In this context, some researchers developed some theories to introduce the concept of graph and causality in feature selection. Their objective is to find the graph used to generate the data with the purpose to have a better understanding of all the underlying mechanisms behind the dataset. This is a crucial operation in biology to find the responsible genes of a particular disease in order to develop specific drugs to cure the patients.

In some situations, the reference population does not have the same distribution as the test population on which we want to predict the target. These variations of distribution can be caused by manipulations of certain features such as gene knock out,etc. . These interventions [1] have a really bad impact on the predictive performance of the model learned on the reference population because a manipulation disconnects the variable from the target and drastically decreases the amount of information possessed by the manipulated variable and so its predictive performance.

[1]In this thesis, we used interchangeably "manipulation" or "intervention"

In the framework of this thesis, we tried to find an optimal subset for predicting the target of a manipulated dataset while learning the model on an unmanipulated dataset. The situation can be describe as follows. We have a source dataset assumed to be non manipulated where all inputs and the output variable are measured and a second target dataset where some inputs are manipulated but only the inputs are measured. We called the source dataset : the observational dataset [2]. While the second one is called the experimental dataset since there are some external interventions.

Therefore, the objective is to exploit the information of the experimental dataset even without the target to find a good set of features with only highly predictive variables for the second dataset. The first step was to approximate the Markov Boundary (Definition 10) of the target in the observational dataset with some causal discovery algorithm from the literature. And then trying to remove all the non-parent manipulated variables to obtain an optimal subset of features. All these notions are explained later in the thesis.

The Lucas example (from [10]) is a perfect example of the situation explained previously. This example is about predicting if a patient is suffering from lung cancer based on eleven inputs features. The graph in Figure 1.1 shows all the causal pathway entailed by the distribution between the eleven variables and the target. In this example the Markov Boundary is composed of these features : Smoking, Genetics, Fatigue, Coughing and Allergy. If there is no manipulations, the problem is to find an optimal subset independently of the nature of the different variables with classical feature selection. In this case, the optimal subset is the Markov Boundary.

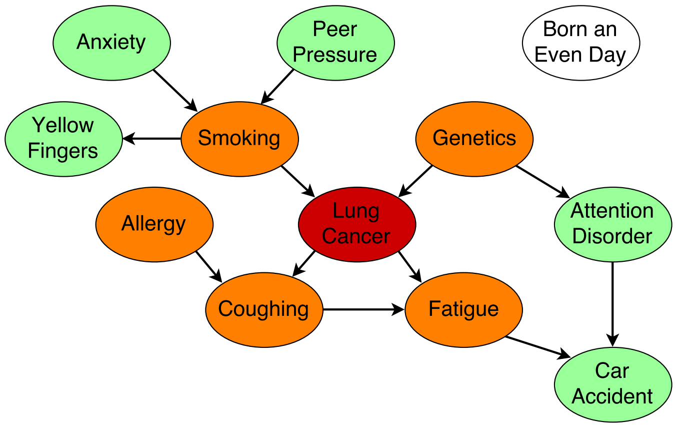
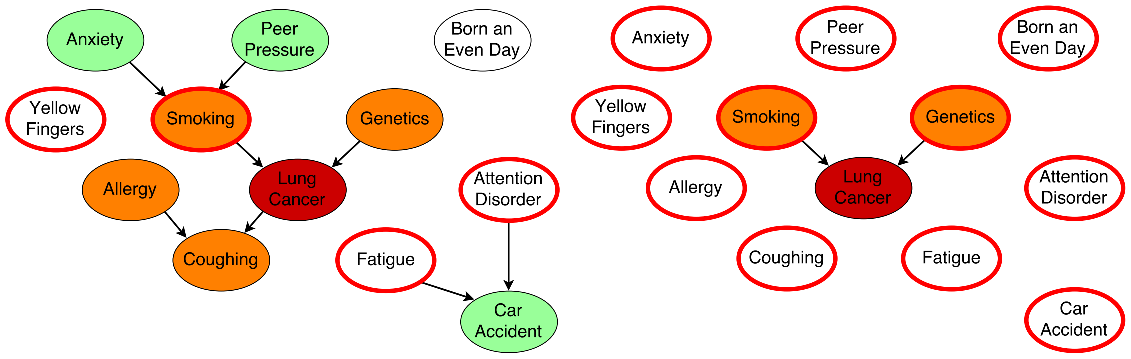


Figure 1.1: Graph of the LUCAS dataset showing the edges between the different features when there is no manipulation. The red, orange, green and white colors respectively represent the target, the Markov Boundary, redundant features and independent features.

If some manipulations are introduced, classical approach will not be enough to find the optimal subset. Indeed, classical approach are likely to output the Markov Bound-

[2]We also used the term learning dataset to describe the observational dataset. When we had to test our models on a dataset drawn from the source distribution, we called it the observational test set

ary or a very close subset but in LUCAS1 (Figure 1.2(a)) we need to remove the variable Fatigue from the Markov Boundary since it is disconnected from the target. In LUCAS2 (Figure 1.2(b)), all the features are manipulated and only the parents remain predictive.



(a) Graph of the dataset LUCAS1 (b) Graph of the dataset LUCAS2

Figure 1.2: Graph of the LUCAS dataset showing the edges between the different features for two different sets of manipulations. The red, orange, green and white colors respectively represent the target, the Markov Boundary, redundant features and independent features. The features circled by a red line represent the manipulated features.

The final objective of this thesis is to find an optimal subset of features to predict the manipulated dataset based on the observational dataset.

1.2 Structure of the report

The chapter 2 draws the state of the art in feature selection. In particularly, we reminded the different notions about classical feature and we introduced the main theories about causal discovery and causal feature selection. Furthermore, this chapter presents the three different families of classical methods for feature selection. A brief description of the two local causal discovery algorithms used in this thesis is also presented.

The chapter 3 describes the datasets used in the framework of this thesis. Since one of the dataset is part of a challenge, we briefly discussed the context and the goal of the challenge.

The chapter 4 presents the different steps followed in the results chapters and describes the two algorithms developed during this thesis : DPED and DCED. The DPED algorithm tries to take advantages of the variations of prediction between two datasets with different distribution while DCED searches for correlation variation around the target. The intuitions and the assumptions made for these algorithms are described and their limitations are highlighted.

The chapter 5 and 6 mainly focus on the results with different approaches including a classical and a causal analysis. We first compared the efficiency of the classical feature selection techniques and the causal discovery algorithms to find an optimal subset of features for prediction on the observational test set. Then we highlighted the limitations of these techniques in a manipulated environment before showing the capabilities of the algorithms developed during this thesis (DPED and DCED) to find subsets with optimal predictive performance on manipulated datasets.

1.3 Notations

In this thesis, we used some specific terms that we redefined in this section.

The nature of a feature is the nature of its connection with the target : parents if it is a direct cause of the target, children if it is a direct effect of the target, spouses if it is a parent of a children of the target. We also distinguish the manipulated and the unmanipulated features. For example, if a direct cause of the target is manipulated, the nature of this variable is manipulated parent. We used the expression external variable to describe a variable which is not a member of the Markov Boundary.

We considered independent variable with respect to the target as irrelevant variable because they are not connected with the target and therefore do not contain any information about the target.

We also called the variables not included in the unique Markov Boundary as external variable.

Chapter 2

Introduction to classical and causal feature selection

Nowadays, the number of features does not stop increasing due to an improve of data acquisition techniques, for example high-throughput experiments in the case of biology. This increase has a huge impact on the model as well as in terms of time consumption or predictivity. Indeed, contrary to common beliefs , more variables is not synonymous of more useful information and a better prediction while in theory the more features are used the better. This can be easily explained by the fact that non relevant features induce overfitting and so decrease the performances and the generalization of the model. Therefore, it is important to reduce the number of features for several reasons such as decreasing the size of the dataset or improving the performances of the prediction.

This chapter summarized the basic principles of feature selection in a data base for both classical and causal techniques. We described the main methods for classical feature selection. We also introduced two local causal pathway discovery algorithms used in the framework of this thesis : HITON and IAMB.

2.1 Classical feature selection

2.1.1 What is classical feature selection

This definition describes simply and succinctly the concept of feature selection : "The variable selection refers to the problem of selecting input variables that are most predictive of a given outcome" [13]. Or in other words, classical feature selection is the process of selecting a subset of features that are the most relevant for predicting the target. It differs from a dimensionality reduction since feature selection attempts to remove unnecessary features while dimensionality reduction tries to reduce the number of features by combining them. The learning algorithm has to select a subset of features upon which to focus its attention while ignoring the rest [14].

Many machine learning algorithms are not well designed to handle large amount of variables and are subjected to overfitting. Since, the attributes of an instance is generically obtained by brute force, the probability to include non relevant features in the data set is quite high. The brute force process describes the fact that one will try to collect all the possible attributes to define one instance. With the evolution of technologies, the number of possible attributes increases especially in biology where feature selection is vital.

A second reason to remove some features is to decrease the cost of observations. For example, in medicine, examination such as gene analyzing may cost a lot of money and lead to a waste of resources if the variable extracted from this examination is not useful for the prediction of the target. Moreover, these examinations are not necessary harmless for the patient.

The last reason of feature selection is related to the interpretability of the variables which is more difficult for researchers if there are some unnecessary variables. This point will be more discussed in the causal feature selection section.

This thesis being mainly focused on supervised inductive classifier machine learning algorithms such as Trees algorithm (RandomForest) or support vector machines (SVM), a more formal definition can be made :

Definition 1. Given an inducer $\mathcal{I}$, and a dataset $\mathcal{D}$ with features $X_1, X_2, ..., X_n$, from a distribution D over the labeled instance space, an optimal feature subset, $\mathcal{X}_{opt}$, is a subset of the features such that the accuracy of the induced classifier $\mathcal{C} = \mathcal{I}(\mathcal{D})$ using only $\mathcal{X}_{opt}$ is maximal [14].

This definition takes into account the heuristics, biases and tradeoffs of the particular inducer $\mathcal{I}$ to define the optimal subset of features. One of the problems of this definition is the computation of the accuracy of the induced classifier because one does not have access to the distribution D and must estimate the classifier's accuracy from the data.

Some feature selection models treat the variable separately and try to remove the independent features based on a ranking. The problem arising with this method is linked to redundant and correlated features. Indeed, some variables contains the same amount of information and only one should be taken but with individual selection, one cannot distinguish between two variables with the same score. This approach keep more features than necessary and so lead to more overfitting.

2.1.2 Relevancy and Redundancy

The relevancy notion is the core of feature selection and has to be well defined to avoid misunderstanding on the purpose of each algorithm. The goal of a feature selection model is to keep the relevant features and remove the non relevant feature but this, ideally, has to be done independently of the metric used to compute the score of a feature or the learning model and should be only dependent on the probability distribution of the data. In some cases, it is more useful to consider the learning model to improve the effectiveness of its prediction but ,in that case, interpretation of the selected variables is biased because of the particularities of the concerned learning algorithm (e.g. a feature can be useful for the prediction even if it is not strongly relevant).

Through this section, a simple example will be used to illustrate the different notions : Let features $X_1, ..., X_5$ be continuous. The target is $T = X_1 + X_2$ with $X_3 = -X_2$ and $X_4 = 2X_5$. The feature X_1 and one of X_2 or X_3 are essential while X_4 and X_5 are useless to determine the target. There are two optimal subsets in this example : $\{ X_1, X_2 \}$ or $\{ X_1, X_3 \}$.

2.1.2.1 Individual Relevancy

The simplest way to define relevancy is to consider the dependencies between an individual variable and the target. If a feature is not correlated to the target, it is considered as irrelevant. A simple definition can be given by :

Definition 2. The feature X_i is individually irrelevant to the target T *iff* X_i is independent of T (denoted as $X_i \perp T$): $P(X_i, T) = P(X_i)P(T)$[8].

With that definition, the relevant features are the complementary of the irrelevant features or, in other words, the variables which depends on the target. However, in practice, the dependency between an individual feature and the target is computed with a statistical test based on a finite number of training samples. Two notions are commonly used in that purpose : The sensitivity and the specificity.

$$sensitivity = \frac{\sum \text{True positive}}{\sum \text{Labeled positive}} \tag{2.1}$$

$$specificity = \frac{\sum \text{True negative}}{\sum \text{Labeled negative}} \tag{2.2}$$

These statistics vary from one statistical test to an other and must be assessed. Since various tests offer tradeoffs between sensitivity and specificity, it must be chosen carefully depending on the application.

From that definition, a feature can be classified in only two categories : relevant or irrelevant. With individual relevancy, the optimal subset in the example is $\{ X_1, X_2, X_3 \}$ because both X_2 and X_3 are correlated with the target even if only one is enough to predict it. To discard one of them, we need to introduce a more complex and complete concept of relevancy : multivariate relevancy and redundancy.

2.1.2.2 Multivariate Relevancy

In this section, we extend the definition of relevancy to include the impact of other features on the relevancy of one feature.

In literature, the relevancy of a feature is often classified in three distinct categories : strongly relevant, weakly relevant and irrelevant. Let T be the class label, $\mathbf{X}$ a full set of features with X_i a feature, $\mathbf{S}_i = \mathbf{X} - \{X_i\}$ and $P(T|\mathbf{X})$ is the conditional class distribution given the values of the features in $\mathbf{X}$, the different notions of relevancy can be defined as follows [18] :

Definition 3. A feature X_i is strongly relevant *iff* :

$$P(T|X_i, \mathbf{S}_i) \neq P(T|\mathbf{S}_i) \tag{2.3}$$

Definition 4. A feature X_i is weakly relevant *iff* :

$$P(T|X_i, \mathbf{S}_i) = P(T|\mathbf{S}_i) \tag{2.4}$$

and $\exists \mathbf{S}'_i \subset \mathbf{S}_i$, such that $P(T|X_i, \mathbf{S}'_i) \neq P(T|\mathbf{S}'_i)$

Definition 5. A feature X_i is independent *iff* :

$$\forall \mathbf{S}'_i \subseteq \mathbf{S}_i, P(T|X_i, \mathbf{S}'_i) = P(T|\mathbf{S}'_i) \tag{2.5}$$

If the feature affects the conditional class distribution, it is labeled as strongly relevant and is always necessary for an optimal subset. Weak relevancy suggests that the feature is not always necessary but may become necessary for an optimal subset at certain conditions. An irrelevant variable is unnecessary and should not be included in the optimal subset. A feature selection model attempts to find the strong relevant features and some of the weak relevant to build the optimal subset. Without defining redundancy, it is hard to understand which weak relevant features must be retrieved while discarding the others.

Feature redundancy is directly linked to feature correlation which means that the information carried by the features are the same and only one is enough to learn this information. This problematic can be quite tricky and required a more complex model such as causal feature selection and dedicated theory like the Markov Blanket which will be discussed later.

The different notions of relevancy can be identify by four categories: strongly relevant(**IV**), weakly relevant but non-redundant features(**III**), redundant(**II**) and irrelevant(**I**) as shown in Figure 2.1. An optimal subset is made of the part (**IV**) and (**III**). The parts (**II**) and (**III**) are not static and the features can be swapped between both categories. For example, two features perfectly correlated will be split and one will be in (**II**) and the other one in (**III**) but they can be in either part without distinction as long as they are not both in the same set.

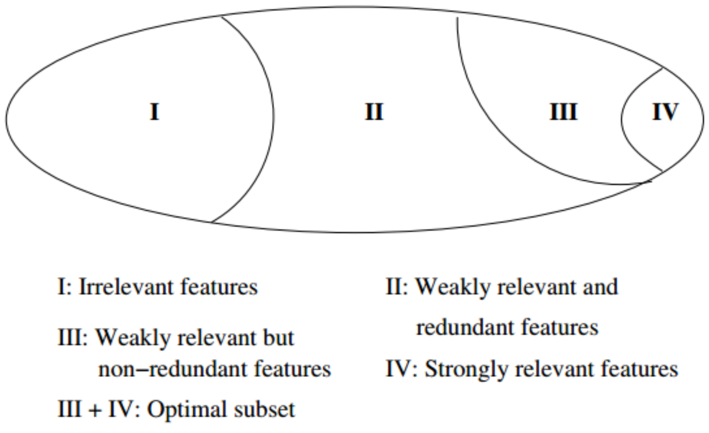

I
II
III
IV
I: Irrelevant features
II: Weakly relevant and redundant features
III: Weakly relevant but non−redundant features
IV: Strongly relevant features
III + IV: Optimal subset

Figure 2.1: A view of feature relevancy and redundancy[18]

In the previous example, X_1 is strongly relevant(**IV**) while X_4 and X_5 are irrelevant (**I**). Either of X_2 or X_3 should be labeled as weakly relevant but non redundant (**II**) and the other as weakly relevant but redundant (**III**) so the optimal subset is either $\{ X_1, X_2 \}$ or $\{ X_1, X_3 \}$ and correspond to the correct optimal subset.

2.1.3 Filter, Wrapper and Embedded method

The feature selection methods can be divided in three categories [4] :

- Filter methods : These methods are independent of the learning algorithm and typically provide a ranking of the features, e.g. based on computing correlation with the output.
- Wrapper methods : Wrappers take into account the learning algorithm to find the best subset of features over a finite space of subsets. In order to do that, the method learns a model for each subset and associates a score to each subset based on the prediction accuracy.
- Embedded methods : They are part of the learning algorithm and perform feature selection during the fitting. It is a black box during the process so the user do not have access to the selected features but the user can get the ranking of the features after the fitting. The Random Forest algorithm is a typical example of embedded method.

Filter methods dissociate the feature selection and the learning algorithm allowing the selection to be more general and computed only once for various classifiers. This family of methods is easily scaled to high-dimensional dataset due to simple and fast computation [4]. The dissociation between the classifier and the feature selection can be detrimental because the interaction with the classifier is ignored. Low computational costs are always reached because these methods are univariate or low-variate which means that each feature is considered separately which may lead to worse performance because of the weak relevant and redundant features (**II**). In the framework of this thesis, we focused our attention on three particular univariate filters : t-test, Pearson and Spearman correlation.

On one hand, wrapper methods integrate the learning algorithm in the feature selection process and take advantage of the interaction between the features and the classifier. On the other hand, they are computationally intensive especially if the learning algorithm has a high building cost.

Embedded methods are build-in methods and are therefore less computationally intensive than wrapper methods but they are specific to a learning algorithm.

In conclusion, all these three methods offer various tradeoffs and can be combined to increase the efficiency of the whole features selection process.

2.2 Causal feature selection

In this thesis, we use the expression "causal discovery" to talk about any algorithm trying to take advantages of the underlying causal mechanisms to find a particular subset of features(e.g. Markov Boundary). It means that we considered algorithms which are looking for the Markov Boundary as causal even if they do not attempt to direct the edges.

2.2.1 Definitions and theorems

In this section, we describe the basic definitions and theorems about causal discovery. We limited our analysis to the concepts involved in the two algorithms described in the next sections because this is a very broad subject and covering all the aspects goes beyond the scope of this thesis.

As a reminder, a direct acyclic graph (DAG) is a finite directed graph with no directed cycles. In a DAG, a node A is the parent of B (B is the child of A) if there is a direct edge from A to B.

Definition 6. Let $\mathbb{G}$ be a direct acyclic graph (DAG) with vertex set $\mathbf{V}$ and J be a probability distribution over the vertices in $\mathbf{V}$ generated by the causal structure represented by $\mathbb{G}$. $\mathbb{G}$ and J satisfy the **Causal Markov Condition** if and only if for every X in $\mathbf{V}$, X is conditionally independent of $\mathbf{V} \backslash (\mathbf{Descendants}(X) \cup \mathbf{Parents}(X))$ given **Parents**(X) [11].

Definition 7. Bayesian Network $<\mathbf{V},\mathbb{G},J>$. Let $\mathbf{V}$ be a set of discrete variables and J be a joint probability distribution over all possible instantiations of $\mathbf{V}$. Let $\mathbb{G}$ be a directed acyclic graph over a set of variables $S \subset \mathbf{V}$. Let all nodes of $\mathbb{G}$ correspond one-to-one to members of $\mathbf{V}$. We require that for every node $V \in \mathbf{V}$, V is probabilistically independent of all non-descendants of V, given the parents of V (**Markov Condition**). Then we call the triplet $<\mathbf{V},\mathbb{G},J>$ a Bayesian Network (BN) [13].

This condition translates the fact that every node is only conditioned by its parents.

Definition 8. DAG-faithfulness: A Directed Acyclic Graph $\mathbb{G}$ is faithful to a joint probability distribution J over a set of variables $\mathbf{V}$ *iff* every independence present in J is entailed by $\mathbb{G}$ and the Markov condition, that is ($\forall A \in \mathbf{V}$; $\forall B \in \mathbf{V}$ and $\forall \mathbf{C} \subset \mathbf{V}$), $A \not\perp_{\mathcal{G}} B|\mathbf{C} \Rightarrow A \not\perp_{\mathcal{J}} B|\mathbf{C}$ [8].

When the objective is to select some important features around the target, the notions of Markov Blanket and Markov Boundary are the most interesting in causal discovery because they offer subsets of features containing a lot of information about the target.

Definition 9. Markov blanket: A Markov blanket $\mathbf{M}$ of the response variable T$\in$ $\mathbf{V}$ in the joint probability distribution J over variables V is a set of variables conditioned on which all other variables are independent of T, that is, for every $\mathbf{X} \in (\mathbf{V} \backslash \mathbf{M} \backslash \{T\}), T \perp \mathbf{X}|\mathbf{M}$ [16].

Trivially, the set of all variables $\mathbf{V}$ excluding T is a Markov blanket of T. From any small Markov Blanket, one can add arbitrary (predictively redundant or irrelevant) variables to generate a larger Markov Blanket. Hence, only minimal Markov blankets are of interest.

Definition 10. Markov boundary: If no proper subset of $\mathbf{M}$ satisfies the definition of Markov blanket of T, then $\mathbf{M}$ is called a Markov boundary of T [16].

In other words, the Markov boundary $\mathbf{M}$ is a minimal set of variables conditioned on which all the remaining variables in the data set, excluding the response variable T, are rendered statistically independent of the response variable T [16]. This set of variables is the minimal set with optimal predictive performance.

Theorem 1. If a joint probability distribution J over variables $\mathbf{V}$ satisfies the faithfulness property, then for each X $\in$ $\mathbf{V}$, there exists a unique Markov boundary of X [16].

Furthermore, from Definition 10 and Theorem 1, we can derive this theorem :

Theorem 2. If a joint probability distribution J is DAG-faithful to $\mathbb{G}$, then the set of children, parents, and spouses of T is a unique Markov boundary of T [16].

The faithfulness between the distribution and the DAG is one of the core notion for a lot of causal discovery algorithms. If the faithfulness is violated, it is possible to find two variables with exactly the same behaviors that cannot be distinguished and therefore cannot be correctly processed.

We introduced the notion of redundancy in the last section without any proper definition. From Definition 9 of the Markov Blanket, we can define a redundant feature :

Definition 11. Redundant feature : Let $\mathbf{X}$ be the current set of features, a feature is redundant and hence should be removed from $\mathbf{X}$ *iff* it is weakly relevant and has a Markov blanket $\mathbf{M}_i$ within $\mathbf{X}$ [18]

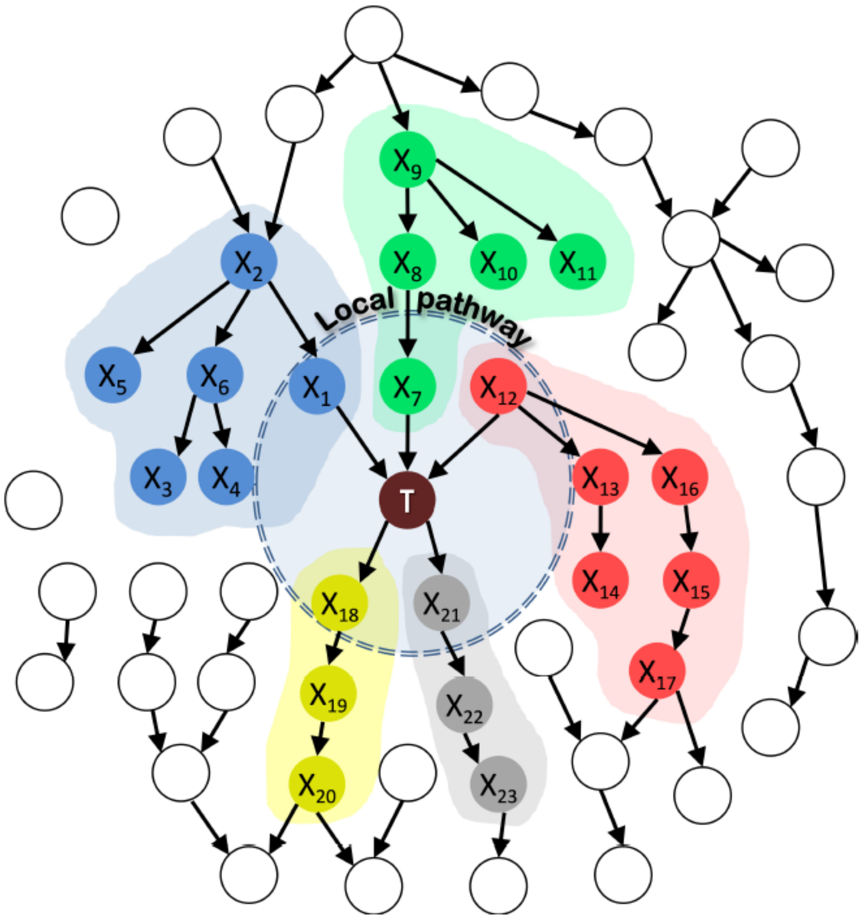


Figure 2.2: Graph around the target for the P1000 dataset [15]

When the graph faithfulness assumption is violated, algorithms for discovery of local causal pathway members from observational data may include false positives in their output while missing some true positives. Consider an example of causal network in Figure 2.2 where the distribution and the graph are not faithful. Indeed, all the variables in the same color contain equivalent information about the target T which violates the assumption made for the faithfulness between the distribution and the graph. The two set $\{X_1\ , X_7, X_{12}, X_{18}, X_{21}\}$ and $\{X_2\ , X_8, X_{13}, X_{19}, X_{22}\}$ are equivalent considering their information about the target T but only one is equivalent to the unique Markov Boundary. Local causal discovery algorithms that assume faithfulness can output any of them. In this example, there are 1620 equivalent set of variables in terms of information about T and each of these sets can be arbitrarily output by a local causal discovery algorithm. Therefore, there is a multiplicity of local causal pathways consistent with the data [15]. The impact of this multiplicity on prediction is limited since only the information is important and not the exact nature of the features if there is no manipulation. On the other hand, if a children of a children of the target is selected in the final output as member of the Markov Boundary and if the children of the target is manipulated, the information about the target contained by the children of the children is very low. When this variable is not manipulated, we will tend to keep it since it is labeled as an unmanipulated member of the Markov Boundary. For example, in Figure 2.2,if we select the variable 22 instead of 21 because of the multiplicity of local causal pathway when the variable 21 is manipulated, the variable 22 should be removed even if this variable is unmanipulated.

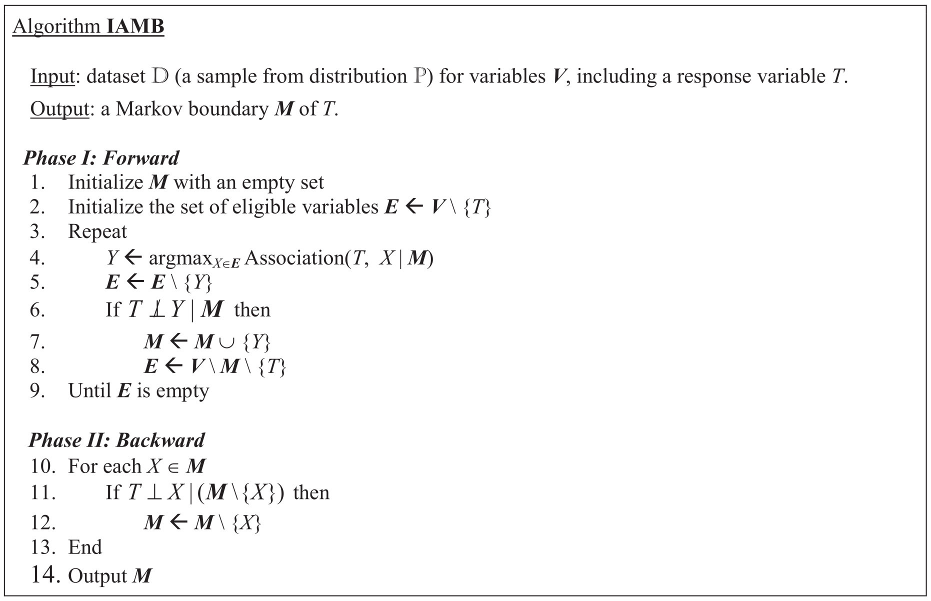
2.2.2 IAMB algorithm

This algorithm is explained in [13] and [17] and implemented in the Causal Explorer Toolkit [1]. IAMB stands for Incremental Association Markov Boundary and is divided in two phases : a forward and a backward one. The goal of this algorithm is to find the Markov Boundary of a target variable T. In the forward phase, a set of features (CMB) is filled with all the variables in the Markov Boundary of T and possibly more variables called false positives. The backward phase focus on removing all the false positives in CMB while keeping all the true positives. The output is the final set found at the end of the backward phase.

The two major assumptions to have a good behavior of IAMB are described below:

- DAG faithfulness of the graph used to generate the data
- Reliable statistical conditional independence tests and measures of associations for checking independence and strength of association of T with some other variable X given a set of variables **Y** [13]

A pseudo code of the IAMB algorithm is shown in Figure 2.3 emphasizing the two phases. We can notice that the conditional independence test in line 6 and 11 is done by conditioning on the entire temporary Markov Boundary (CMB).



```
Algorithm IAMB

  Input: dataset $\mathbb{D}$ (a sample from distribution $\mathbb{P}$) for variables $\boldsymbol{V}$, including a response variable $T$.
  Output: a Markov boundary $\boldsymbol{M}$ of $T$.

Phase I: Forward
 1.  Initialize $\boldsymbol{M}$ with an empty set
 2.  Initialize the set of eligible variables $\boldsymbol{E} \leftarrow \boldsymbol{V} \setminus \{T\}$
 3.  Repeat
 4.      $Y \leftarrow \text{argmax}_{X \in \boldsymbol{E}} \text{Association}(T, X \mid \boldsymbol{M})$
 5.      $\boldsymbol{E} \leftarrow \boldsymbol{E} \setminus \{Y\}$
 6.      If $T \perp Y \mid \boldsymbol{M}$ then
 7.          $\boldsymbol{M} \leftarrow \boldsymbol{M} \cup \{Y\}$
 8.          $\boldsymbol{E} \leftarrow \boldsymbol{V} \setminus \boldsymbol{M} \setminus \{T\}$
 9.  Until $\boldsymbol{E}$ is empty

Phase II: Backward
 10. For each $X \in \boldsymbol{M}$
 11.     If $T \perp X \mid (\boldsymbol{M} \setminus \{X\})$ then
 12.         $\boldsymbol{M} \leftarrow \boldsymbol{M} \setminus \{X\}$
 13. End
 14. Output $\boldsymbol{M}$
```

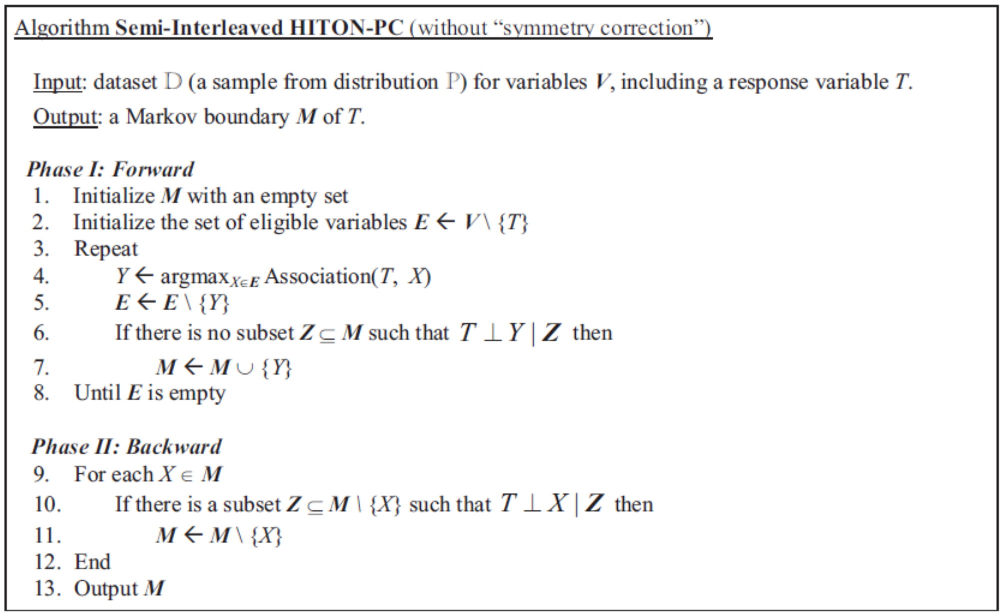
Figure 2.3: Pseudo code of the IAMB algorithm [16]

A statistical threshold α allows the user to have a control on the number of included features.

2.2.3 HITON algorithm

The HITON algorithm was first introduced in [2] and then redefined in [3]. The Hiton algorithm has the same well-specified set of assumptions for inducing the Markov Boundary including faithfulness between the graph and the distribution and reliable statistical tests of independence.

This algorithm mainly differs from IAMB in the conditioning set of variable while testing the conditional independence of the variable Y (line 6) or X(line 10) in Figure 2.4). Reducing the size of the conditioning set is really important for discrete data where the sample size required for high-confidence statistical tests of conditional independence grows exponentially in the size of the conditioning set [16].



```
Algorithm Semi-Interleaved HITON-PC (without "symmetry correction")

  Input: dataset D (a sample from distribution P) for variables $V$, including a response variable $T$.
  Output: a Markov boundary $M$ of $T$.

Phase I: Forward
1.  Initialize $M$ with an empty set
2.  Initialize the set of eligible variables $E \leftarrow V \setminus \{T\}$
3.  Repeat
4.      $Y \leftarrow \mathrm{argmax}_{X \in E}$ Association($T$, $X$)
5.      $E \leftarrow E \setminus \{Y\}$
6.      If there is no subset $Z \subseteq M$ such that $T \perp Y \mid Z$ then
7.          $M \leftarrow M \cup \{Y\}$
8.  Until $E$ is empty

Phase II: Backward
9.  For each $X \in M$
10.     If there is a subset $Z \subseteq M \setminus \{X\}$ such that $T \perp X \mid Z$ then
11.         $M \leftarrow M \setminus \{X\}$
12. End
13. Output $M$
```

Figure 2.4: Pseudo code of the HITON PC algorithm [16]

The implementation of the HITON algorithm used in the framework of this thesis come from the Causal Explorer Toolkit [1]. There are two derived algorithms from the HITON algorithm : HITON pc and HITON MB. The first one only tries to find the parents and children of the target while the other tries to also include the spouses.

A statistical threshold α allows the user to have a control on the number of included features.

Chapter 3

Datasets

3.1 Causality Challenge

In this section, we introduced the causality and prediction challenge from WCCI 2008 [10] from which most example are drawn. This challenge is focused on feature selection to predict a database manipulated by an external agent with a model learned on an unmanipulated dataset but also on discovering underlying mechanisms linking the different variables. In the framework of this challenge, we focused on a local causal discovery around the target without attempting to uncover the full graph behind the database. It is important to take care of causality when there are some manipulations on the variables in the test set because these manipulations remove any correlation between the features and the target and must be taken into account when selecting predictive features.

There are some important fields where causal feature selection is vital to obtain good results for both prediction and comprehension of the phenomena such as :

- Medical domain : In the case of a cancer, we need to know what are its causes and on which variables do we need to act to cure it. For example, in lung cancer, we have smoking and coughing as variables and we need to know which is a cause and a consequence because treating coughing which is a consequence will not cure the cancer while prohibiting smoking may prevent the cancer because it is a direct cause.
- Econometrics : Investors want to know the effect of a new policy before investing.

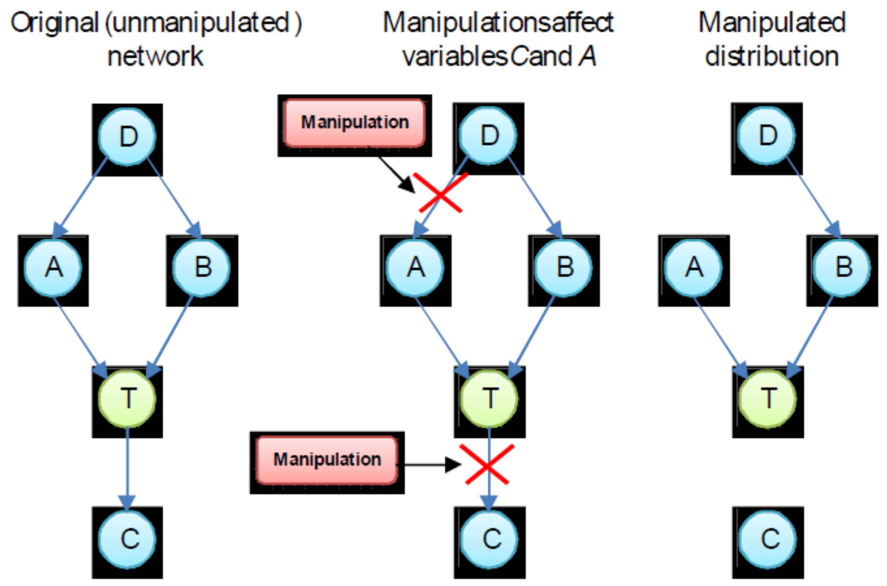
In order to explore all the possibilities, four families of datasets are provided by the organizer : REGED, SIDO, CINA and MARTI. This thesis only focus on the REGED family which is described in Section 3.2. In the REGED family, one learning set and three test sets are provided : REGED LS, REGED0, REGED1 and REGED2.

3.2 REGED

REGED is a re-simulated dataset and stands for **Re**simulated **G**ene **E**xpression **Dataset**. A simulator of gene expression data trained on real DNA microarray data is used to generate REGED. The data are gene expression while the target is lung cancer subtype. The target is binary and separates two subtypes of cancer. The task is to discover genes which allow a good prediction on the subtype of cancer. In medical domain, there are multiple external agents acting on the expression of the genes. Some manipulations were introduced to simulate the effect of agents such as drugs or gene knockout (when a gene is suppressed). We called observational the dataset without any external manipulations and experimental the dataset with manipulations. In REGED, the training set and the test set REGED0 are drawn from an observational distribution while REGED1 and REGED2 are drawn from an experimental distribution. On the observational test set REGED0, the task is to discover genes which directly impact (causes) or are directly impacted (effects) by the disease because this set of genes is optimal for prediction. When there are some manipulations in the test set (REGED1 and REGED2), the task is to find the genes that are still predictive in the experimental test set after manipulations based on the observational learning set.

All the datasets include 999 continuous features describing the genes expression. The data are preprocessed to obtain features in the same discrete numerical range [0,999]. The learning dataset is made of 500 training samples drawn from the unmanipulated distribution. The training set is the same for the three test set independently of the manipulations in the different test sets. There are 20000 samples in each test set. An example of manipulation and its impact on the set of predictive variables is shown in Figure 3.1.

In this thesis, we used the expression "observational dataset" to describe the learning set and the expression "experimental dataset" to describe both REGED1 and REGED2. For REGED0, we used observational test set.


Original (unmanipulated) network
Manipulations affect variables C and A
Manipulated distribution
Manipulation
Manipulation
D
A
B
T
C
D
A
B
T
C
D
A
B
T
C
Figure 3.1: Example of the effects of manipulations. T is the target (response) variable. In the original network, both A (direct cause of T) and C (direct effect of T) are predictive of T. However, once C and A are manipulated, only A remains predictive of T in the manipulated network while B remains predictive in both distribution.[9]

The test sets REGED1 and REGED2 are drawn from manipulated distributions. In order to do that, the network graph is manipulated and then the network is re-parameterized accordingly and data (20 000 samples) is generated from it. This implies that the manipulations are propagated in the distribution allowing the parents to be still predictive even if they are manipulated what would not have been possible if the manipulations were done directly after the sampling on the genes expression. Here are the details of the manipulations for both REGED1 and REGED2 :

- REGED1 : 1 direct cause of the target, 5 of its mostly predictive direct effects and 94 randomly selected variables, which do not belong to the local neighborhood of the target, are manipulated for a total of 100 manipulated variables
- REGED2 : 1 direct cause of the target, all (13) of its direct effects and 86 randomly selected variables, which do not belong to the local neighborhood of the target, are manipulated for a total of 100 manipulated variables

In the case of REGED1, the organizers provided the list of manipulated features : 83, 251, 321, 409, 593, 939 for the local neighborhood. We also had access to the list of the 94 randomly selected variables.

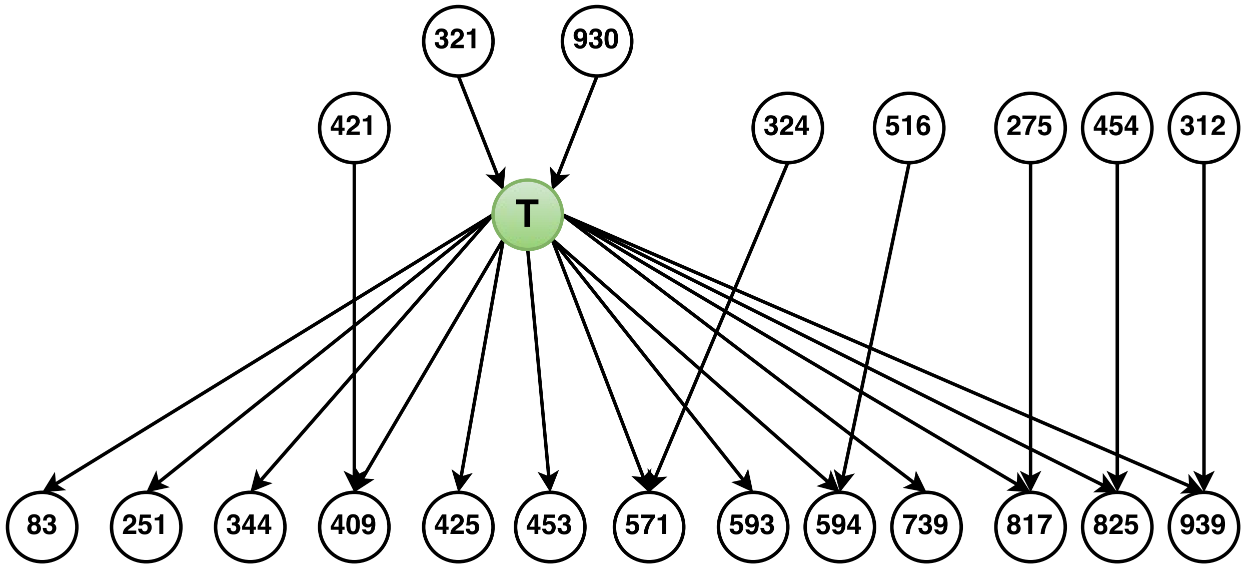


Figure 3.2: Local graph of T in the REGED network

In Figure 3.2, we can see the local neighborhood of the target T in REGED network. The features are numbered from 1 to 999. This set of features corresponds to the Markov Boundary of T if the graph faithfulness is respected. We tried different approaches to find this subset of features in Chapter 5.

In order to assess the efficiency of the prediction on the three datasets, a score is computed for each prediction. The score used in the framework of this challenge is the area under the ROC curve which is identical to the balanced accuracy (BAC) in this particular case:

$$BAC = \frac{sensitivity + specificity}{2} \tag{3.1}$$

The definitions of sensitivity and specificity are described in equations 2.1 and 2.2.

We computed some reference scores to be able to compare the results presented in Chapter 5 and 6. We used different particular subsets of features :

- $\mathcal{X}_{\text{All}}$: Subset with all the features
- $\mathcal{X}_{\text{MB}}$: Markov Boundary
- $\mathcal{X}_{\text{MB/manip}}$: Markov Boundary without the features manipulated to generate REGED1
- $\mathcal{X}_{\text{Parents}}$: Only the parents of T

We used a Random Forest Classifier implemented in Scikit Learn with 1000 estimators and all the other parameters are left by default. The results are shown in Table 3.1.

	$\mathcal{X}_{\text{All}}$	$\mathcal{X}_{\text{MB}}$	$\mathcal{X}_{\text{MB/manip}}$	$\mathcal{X}_{\text{Parents}}$
Reged0	0.8164	0.9835	0.9314	0.8122
Reged1	0.5114	0.5751	0.9271	0.8079
Reged2	0.5011	0.5140	0.6240	0.8233

Table 3.1: Reference prediction scores for REGED computed with a Random Forest classifier (n_estimators = 1000) and different subsets of features

We can highlight the efficiency of the subset $\mathcal{X}_{\text{MB}}$ on the observational test set REGED0 while removing the manipulated features used to generate REGED1 has huge impact on the score for REGED1. When only the parents are used, the scores are almost the same for all test set. The only significant score for the highly manipulated test set REGED2 is obtained with $\mathcal{X}_{\text{Parents}}$ showing the importance of retrieving the parents in such context.

In Chapter 5 and 6, we used support vector machine which requires a preprocessing of the dataset such as standardizing the features by removing the mean and scaling to unit variance. This is done with the help of the *StandardScaler* function provided by Scikit Learn. We did this preprocessing only for the prediction made with a SVM algorithm.

An important fact is that we assumed the REGED database to be faithful.

3.3 P1000

The P1000 dataset is generated from an artificially simulated network [15]. The graph of the network is manually generated and parameterized using Gaussian distribution. The structure of the dataset is separated in two parts : a causal graph and some independent variables. The causal graph is made of 54 variables including the target while the rest of the 946 variables are drawn independently. The local graph is shown in Figure 2.2. To avoid the problem explained in the last chapter about multiplicity of local causal pathway and to ensure the faithfulness of the graph, we added some noise on every edges.

All the variables are continuous including the target which means that we had to use regressor algorithms instead of classifiers. There are two test set and one learning set. The training set and one of the test are drawn from the unmanipulated distribution and we respectively called them observational learning set and observational test set. The second test set is simulated with a manipulated distribution and we called it experimental test set. We manipulated one parent and the two children as well as nine other variables. The list of the manipulated features is 1, 2, 8, 13, 16, 18, 19, 21, 22, 40, 146, 350.

In regression, one easy way to assess the performance of a prediction is to compute the mean absolute error :

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| \tag{3.2}$$

where f_i is the prediction, y_i is the true value and n the number of samples. We computed some reference prediction errors to be able to compare further results (see Table 3.2). The Markov Boundary is composed of three parents and two children as it can be seen in Figure 2.2.

	$\mathcal{X}_{\text{All}}$	$\mathcal{X}_{\text{MB}}$	$\mathcal{X}_{\text{Parents}}$
Observational test set	0.1103	0.0960	0.0995
Experimental test set	0.7326	0.8032	0.0921

Table 3.2: Reference prediction errors for P1000 computed with a Random Forest Regressor (n_estimators = 1000) and different subsets of features

3.4 Chandran and Singh

These datasets are built from real prostate cancer micro array [12]. The micro array technology is the same for both dataset making possible to apply all the algorithms developed in this thesis. The problem is a classification between two classes : normal and tumor tissues. The type of tumor tissues is not especially the same for the two dataset since, for Chandran, there are two type of tumor samples : primary and metastatic tumor. While for Singh, there is no particular precision about the type of tumor tissue. A brief description of the two dataset is shown in Table 3.3.

dataset	Normal	Tumor	Features
Chandran	18	86	12 625
Singh	50	52	12 625

Table 3.3: First column shows the dataset name, the second and third column describe the number of normal and tumor samples and the last column the number of features

We decided to used Singh as the observational dataset and Chandran as the experimental dataset because of the disparity of the samples for Chandran.

The metric used to evaluate the accuracy of the prediction is the balanced accuracy (see 3.1).

Chapter 4

Algorithms

4.1 Introduction

One of the problems due to interventions is the very poor performance of a model learned on an unmanipulated dataset to predict the target of a manipulated dataset. This effect is increased by the importance of the manipulated variables. Indeed, a manipulation on an important feature has huge impact on the prediction while manipulation on an independent feature has a lower impact. In the case of causal discovery, the difference between manipulated children and manipulated parents must be done because the latter are the only ones still predictive after the manipulations. Indeed, the parents (manipulated or not) always affect the target and so carry the same amount of information before and after manipulation while unmanipulated features (children or others) are still predictive and can be used to learn the model.

These algorithms try to remove all the manipulated features (except the parents) from a subset $\mathcal{X}_S$ previously computed ($\mathcal{X}_S$ is an argument). The variables that we want to remove can be separated in two different groups:

- Manipulated children
- Manipulated variables not in the Markov Boundary

These algorithms were developed in the framework of a more global view about feature selection in a manipulated environment. We proposed a three stages approach to find an optimal subset of features to predict the experimental dataset :

1. Select relevant feature subset $\mathcal{X}_S$ from observational data (eg. Using SVM, Random Forest or causal methods)
2. Filter out lowly predictive features from $\mathcal{X}_S$ such as independent features
3. Filter out manipulated non-parent features from $\mathcal{X}_{\text{S filtered}}$ using both observational and experimental data.

Step 1 can be performed using classical feature selection techniques (eg., SVM or Random Forests) or using causal feature selection techniques.

Step 2 was mainly added to compensate some of the limitations of the algorithms used in step 3.

We will present below two algorithms to perform step 3 :

4.2 DPED Algorithm

4.2.1 Introduction

In a nutshell, the objective is to remove all the manipulated features from a subset of features except the parents (because even manipulated they are still predictive) by comparing the differences between the predictions from models based on single feature in both observational and experimental dataset : Differential Prediction for Experimental Dataset (**DPED**).

The inputs of this algorithm are the observational dataset with the target variable and the experimental dataset, a subset $\mathcal{X}_S$ made of an approximated Markov Boundary of the target previously computed and some parameters described later in the section. The subset $\mathcal{X}_S$ can also be computed from a classical filter ranking by taking the k best features. In that case, the algorithm tries to remove all the non-parent manipulated features but we do not have any information on the nature of the output features.

The algorithm is based on a simple intuition that could be described by a single statement :

Statement :
If two models respectively based on two different features make the same predictions on the observational data set but make completely different predictions on the experimental data set it is likely that one of these two features has been manipulated

This statement was inspired by the fact that a manipulation on a non-parent variable disconnects it from the target and so the prediction made with this feature will be closer to a random assignation than anything else. This variation of the prediction for the same feature before and after a manipulation depends on the manipulation itself. If the manipulation is strong and the feature predictive enough we assume that this statement stands.

To highlight these variations, it is needed to learn models on an unmanipulated database and then predict on both unmanipulated and manipulated dataset. We check if the predictions respectively made using two features are consistent between the observational and experimental datasets. In order to do that, the algorithm takes as argument two datasets (the observational and the experimental). The observational dataset is used as the learning set and must include the target but the experimental dataset is only used to predict so the target is not required. This is a very important characteristic of the DPED algorithm because the target of the experimental dataset is not always so easy to sample. If we had access to a learning set with the target for the experimental dataset, the problem would have been a classical feature selection.

Since this algorithm is used to reject all the manipulated features except the manipulated causes, the unmanipulated features which are not directly connected to the target are kept. If one wants to get only the parents and the unmanipulated children of the target, this algorithm needs to be combined with a causal discovery algorithm like the IAMB or the HITON PC algorithm.

4.2.2 Implementation

The goal of this algorithm is to extract as much information as possible from the two dataset (the observational data with the target and the experimental data without the target). The idea is to compare the predictions made on both datasets with models learned on each feature from the input subset $\mathcal{X}_S$ separately to uncover the manipulations done in the experimental data set. The first step is to fit **n** models with **n** being the number of tested features. Each model is learned on a single feature. The learning algorithm can be modified to fit the data as much as possible. We only explored the Random Forest family in this thesis but DPED can work with any learning algorithm.

In the case of the learning dataset, if we use the same data set to learn the models and predict the target this could lead to a perfect biased score. To avoid that the easiest solution is to implement a k-fold cross validation where the dataset is divided in k subsets. The prediction for each subset is done with a model learned on the k-1 other subsets. We repeat this process for each feature in $\mathcal{X}_S$.

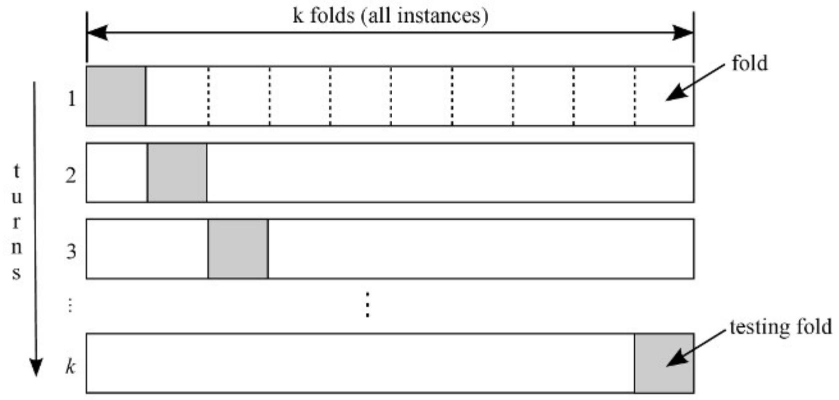


Figure 4.1: Example of a k-fold cross validation

To predict the target for the experimental dataset, a model is learned on the whole observational set for each tested feature. The fitting is really time consuming especially if the number of tested features increases because for each additional feature the algorithm have to fit k+1 models (the k models for the k-fold and one for the experimental data set). By default, the number of fold is ten and could be modified directly in the code but it is not an argument of the final function to avoid an excess of parameters for the user.

At this stage, there are **n** predictions vectors for the learning set and **n** predictions vectors for the experimental set. These arrays can be of different size and a normalization is required to be able to compare them. For example, in the REGED database, the learning sample contains 500 samples but the experimental set contains 20000 samples.

To compare the different prediction from the models learned on a single feature, we introduced an equation which can be seen as an error rate between two predictions (See equation 4.1). The easiest way to compute the error rate is to take the absolute value of the difference between two predictions arrays for some particular features i and j then sum all the values of this array and divide it by the number of samples to normalize the error rate. The main advantage of this error is that we can apply it to both classification and regression. This error is also called the mean absolute error:

$$M_{i,j} = \frac{\sum_{k=1}^{m} |f_i(X^k) - f_j(X^k)|}{m} \tag{4.1}$$

m is the number of samples.
$f_i(X)$ is the model learned on the ith tested feature
$f_i(X^k)$ is the prediction of the kth sample computed with the $f_i(X)$ model

The equation 4.1 is applied to all the predictions arrays of both database (observational and experimental) to generate two n-by-n matrices M_{LS} and M_{TS}. M_{LS} is the error rate matrix for the learning set (also called observational set) and M_{TS} is the error rate matrix for the test set (also called experimental set). The M matrices are symmetrical and the diagonal values are equal to 0 this can be easily noticed from equation 4.1 if $i = j$. The values of these matrices follow different distributions due to the manipulation on some features of the tested subset. The goal is to find these variations and the manipulated features causing them.

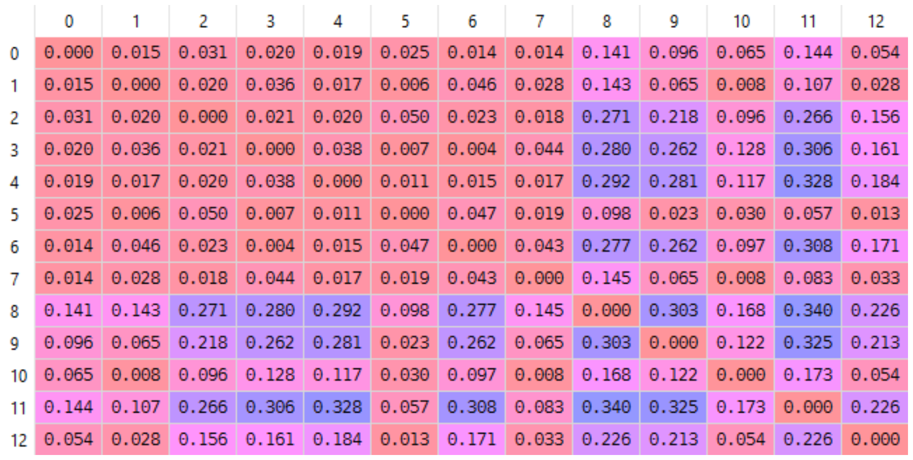
This is simply done by taking the difference between M_{LS} and M_{TS} as shown in equation 4.2.

$$K = |M_{LS} - M_{TS}| \tag{4.2}$$

At this point, the matrix K should include very low values when the predictions done with two different models on the two data base are close and large values when they are very different. If the $K_{i,j}$ value is small, it means that the predictions done with the ith and the jth features are as close in the experimental data as they are in the observational data, which suggests neither feature i, nor feature j are manipulated in the experimental data.

At this stage of the process, we have a n-by-n matrix containing the variations of the error rate between the observational and the experimental dataset with low values for unmanipulated variables and large values for manipulated ones.

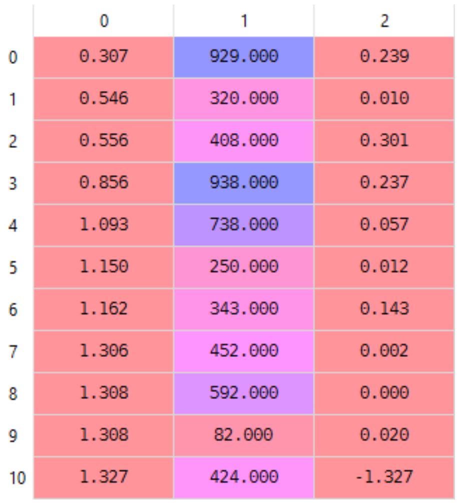
In Figure 4.2, we can see an example of the matrix K from a real run on the REGED dataset and in particular on REGED1. Here, the eight first features corresponds to the good features (parents and unmanipulated children) while the last ones are bad features (manipulated children). We can clearly see the area corresponding to the good features because their values are very low in contrast with differential error rate for the bad features.



	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.000	0.015	0.031	0.020	0.019	0.025	0.014	0.014	0.141	0.096	0.065	0.144	0.054
1	0.015	0.000	0.020	0.036	0.017	0.006	0.046	0.028	0.143	0.065	0.008	0.107	0.028
2	0.031	0.020	0.000	0.021	0.020	0.050	0.023	0.018	0.271	0.218	0.096	0.266	0.156
3	0.020	0.036	0.021	0.000	0.038	0.007	0.004	0.044	0.280	0.262	0.128	0.306	0.161
4	0.019	0.017	0.020	0.038	0.000	0.011	0.015	0.017	0.292	0.281	0.117	0.328	0.184
5	0.025	0.006	0.050	0.007	0.011	0.000	0.047	0.019	0.098	0.023	0.030	0.057	0.013
6	0.014	0.046	0.023	0.004	0.015	0.047	0.000	0.043	0.277	0.262	0.097	0.308	0.171
7	0.014	0.028	0.018	0.044	0.017	0.019	0.043	0.000	0.145	0.065	0.008	0.083	0.033
8	0.141	0.143	0.271	0.280	0.292	0.098	0.277	0.145	0.000	0.303	0.168	0.340	0.226
9	0.096	0.065	0.218	0.262	0.281	0.023	0.262	0.065	0.303	0.000	0.122	0.325	0.213
10	0.065	0.008	0.096	0.128	0.117	0.030	0.097	0.008	0.168	0.122	0.000	0.173	0.054
11	0.144	0.107	0.266	0.306	0.328	0.057	0.308	0.083	0.340	0.325	0.173	0.000	0.226
12	0.054	0.028	0.156	0.161	0.184	0.013	0.171	0.033	0.226	0.213	0.054	0.226	0.000

Figure 4.2: Example of the K matrix for REGED1 with $\mathcal{X}_{\text{hiton}}$ as input subset

After computing the matrix K with the initial **n** features, the objective is to remove the features with the highest values in a clever way. In order to do that, a score is computed for each feature based on the K matrix. The score used in the framework of this algorithm is the sum of the values within the column corresponding to each feature.



	0	1	2
0	0.307	929.000	0.239
1	0.546	320.000	0.010
2	0.556	408.000	0.301
3	0.856	938.000	0.237
4	1.093	738.000	0.057
5	1.150	250.000	0.012
6	1.162	343.000	0.143
7	1.306	452.000	0.002
8	1.308	592.000	0.000
9	1.308	82.000	0.020
10	1.327	424.000	-1.327

Figure 4.3: Example of the features ranking after one iteration for REGED1

The Figure 4.3 shows an example of the features ranking. The first column is the score computed as follows :

$$s_j = \sum_{i \in \mathcal{X}_{DPED}} K_{i,j} \tag{4.3}$$

The second column is the feature's label and the last column is the increment between two consecutive features.

The problem appearing at this stage is the threshold because the user has to define what is a low and a high value of the score to allow the algorithm to identify and then reject the bad features. The first idea was to find the biggest increment between two consecutive scores after sorting all the scores. Indeed, all the good features (unmanipulated features and the parents) should have low scores while manipulated features should have significantly higher scores. Even if one cannot know before all the computations the absolute values of all the scores, the good and bad features should always be separated by a significant increment as it can be seen in Figure 4.3. After multiple tests, this approach was not good enough because there were often more than two distinct areas. For example in Figure 4.3, we can see multiple areas : {929} , {320, 408}, {938}, {738, 250,343} and {452, 592, 82, 424}. In this example we only want to retrieve {929} and {320, 408}. That's why, we implemented an iterative process to remove the last distinct area at each iteration to isolate the good features. At each iteration, a new score array must be computed without the bad features found at the previous iteration and so a new maximal increment by removing the lines and the columns corresponding to the bad features in the matrix K. This process is repeated until the maximal increment reaches a stop criterion ϵ set by the user . This is implemented as the "max increment" mode.

Another idea was to remove the feature with the biggest score and repeat this until the maximal increment decreases below some threshold set by the user. This is implemented as the "last" mode.

A pseudo-code is provided below for the DPED algorithm :

Algorithm 1 DPED

```
Input: observational data $X_{LS}$, observational target $T_{LS}$, experimental data $X_{TS}$,
    tested features subset $\mathcal{X}_S$, learning algorithm, mode, $\epsilon$
Output: $\mathcal{X}_{DPED}$
  $k \leftarrow 10$
  Divide the observational data set into k subsets
  for $i = 0$ to length($\mathcal{X}_S$) do
    Learn the model $f_i$ using ($X_{LS}$,$T_{LS}$)
    Predict the targets of $X_{LS}$ using the k-fold cross validation
    Predict the targets of $X_{TS}$ using $f_i$
  end for
  for $i = 0$ to length($\mathcal{X}_S$) do
    for $j = 0$ to length($\mathcal{X}_S$) do
      Compute $M_{LS_{i,j}}$
      Compute $M_{TS_{i,j}}$
    end for
  end for
  Compute the matrix K
  $\mathcal{X}_{DPED} \leftarrow \mathcal{X}_S$
  Compute the scores of the features in $\mathcal{X}_{DPED}$ with $s_j = \sum_{i \in \mathcal{X}_{DPED}} K_{i,j}$
  Sort the features based on their scores
  Compute the increment between each consecutive features and search for the maxi-
  mum increment
  if mode = "max increment" then
    while max increment $> \epsilon$ do
      Remove the features with higher score than the feature with the max increment
      in $\mathcal{X}_{DPED}$
      Recompute the scores $s_j for j \in \mathcal{X}_{DPED}$
      Recompute a new max increment
    end while
  else if mode = "last" then
    while max increment $> \epsilon$ do
      Remove the feature with the highest score in $\mathcal{X}_{DPED}$
      Recompute the scores $s_j for j \in \mathcal{X}_{DPED}$
      Recompute a new max increment
    end while
  end if
  return $\mathcal{X}_{DPED}$
```

4.2.3 Limitations

Like all algorithms, there are some drawbacks. Their effects can be mitigated by correct settings and good preprocessing. There are two main limitations of this algorithm :

- Lowly predictive variables

- Model learned on a single variable

These limitations are described below.

4.2.3.1 Lowly predictive variables

During the test phase of the algorithm, it appeared that the performance were affected by the nature of the features in $\mathcal{X}_S$. Two different types of feature were problematic : irrelevant features and lowly predictive manipulated features. If there are one or more lowly predictive features in the input subset, they were systematically selected even if they are manipulated because their prediction performance are not particularly modified by the manipulation. This implies that their final score are quite small despite any manipulation. Indeed, since this algorithm is based on differences between predictions, a lowly predictive feature always gives the same kind of prediction independently of its nature (included in the Markov Boundary or not) and manipulations. This explains why the score of such a feature is significantly low and considered as a good feature by DPED. The irrelevant features has the same behavior because their prediction is closer to a random assignation in both dataset independently of the manipulations.

These features are not such a big problem when it comes to prediction because learning a model with the observational dataset on lowly predictive features is not supposed to affect too much the performance if there are enough good (parents and unmanipulated children) and highly predictive features in the final subset and if the chosen learning algorithm is robust with respect to irrelevant features. Indeed, the learning algorithm will automatically give low importance to these features but in the case where lowly predictive and irrelevant features are dominating, poor performances may be expected. Moreover, these features have a really bad impact on the features ranking because they may push out good features from the selected area. For example, such a feature may have a score smaller than the ϵ value provided by the user and so kick out all the good features.

The presence of these features in the tested subset can be limited by a correct Markov Boundary discovery to avoid any error concerning the nature of the features and has to be done through a good selection of the parameters of the algorithm used to discover the Markov Boundary like the threshold for the statistical test in HITON PC or IAMB. Even if the approximated Markov Boundary contains no wrong features (only parents, children and spouses of the target), there are still a probability to have lowly predictive variables that should be removed.

A preprocessing step was essential to remove the lowly predictive manipulated features and the irrelevant features. We implemented a function *feature_filtering* which removes these features in $\mathcal{X}_S$ by selecting the features with a score higher than a threshold set by the user in a Random Forest ranking or a linear SVM ranking. The function compares $\mathcal{X}_S$ and the k more important features from the considered ranking and only keeps the variables present in both subsets. In this context, the most suitable ranking is from learning algorithm such as Random Forest or linear SVM. The model used to compute the score of each feature and the value of the parameter k are provided by the user to allow more flexibility of the code.

4.2.3.2 Models learned on a single feature

To compute the prediction, we only used models learned on a single feature to be able to easily compute a score for each feature. The simplicity of this approach has an important drawback, the prediction may not be good even if the variable used is predictive. For example, let's take X_1,X_2 and X_3 as boolean variables :

$$X_1 = X_2 \oplus X_3 \tag{4.4}$$

In this example, X_2 or X_3 alone will not give significantly good predictions while taken together they perfectly predict the target X_1. To take into account this phenomenon, we should repeat the same procedure for all the possible subset but this approach is too much time consuming and almost intractable.

4.2.3.3 Other limitations

Since this algorithm involves prediction and fitting multiple times, it is quite time consuming. Moreover, the choice of the model and the settings have big impact on the prediction and may possibly affect the final results. For example, Random Forest introduces a lot of randomization and if a feature's score is really close to the threshold, it is possible to observe some variations in the final subset due to the randomization.

4.3 DCED Algorithm

4.3.1 Introduction

In some context, the target can be hard to sample and we could only have observational and experimental data without the target variable but we still want to be able to retrieve the useful features for prediction. Indeed, if the target can be expensive to observe, it is the same for the other variables and reducing the number of selected features is really important especially in biology where sampling can be dangerous for the patient.

The DPED algorithm can be time consuming due to the multiple fitting inherent to the process. The DCED algorithm tries to give fast results based on more simple function to compute the score of each feature for more interpretability.

The inputs of this algorithm are reduced to two datasets (one without manipulations and one with manipulations) without their respective target, a subset made of an approximated Markov Boundary of the target previously computed and some parameters described later in the section.

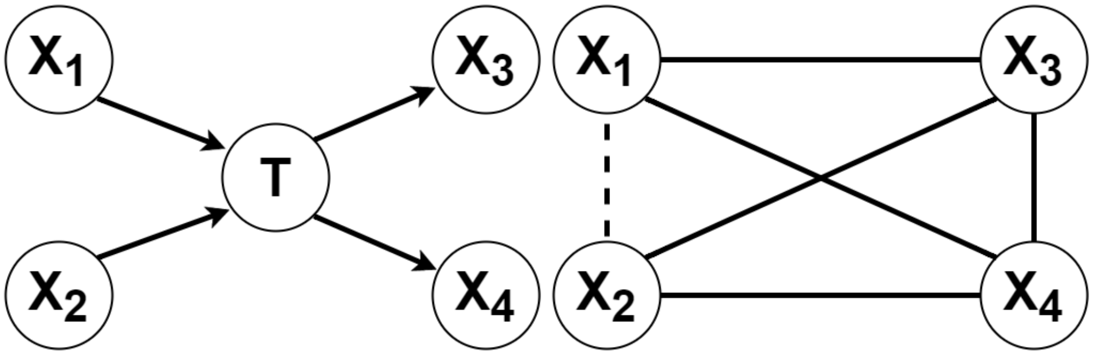
4.3.2 Implementation

The idea behind this algorithm is mainly the same as DPED algorithm but without prediction and without the target. In order to do that, the error rate between the predictions of the variables is replaced by a correlation coefficient. In other words, this algorithm exploits the variation in correlation between variables in the two datasets (observational and experimental). DCED stands for Differential Correlation for Experimental Dataset.

This approach is motivated by the impact of interventions on the correlation between two variables and the simplicity of the evaluation function as well as the interpretation of the results. Indeed, correlation coefficient can help to understand some simple underlying mechanisms. In the framework of this algorithm, we focused our attention on two correlation coefficient : Pearson and Spearman correlation coefficient.

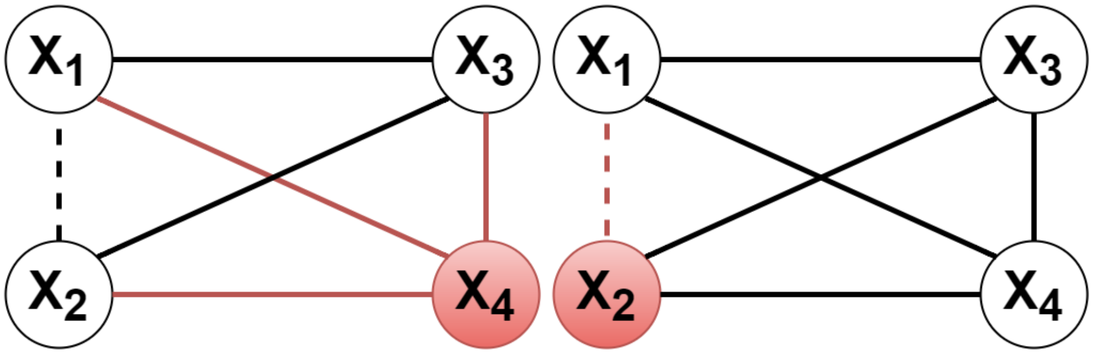
- Pearson : benchmarks linear relationship
- Spearman : benchmarks monotonic relationship

Let's take an example of a small local causal graph around a target T to illustrate the different cases that we have to deal with. The variables X_1 and X_2 are the parents of T while X_3 and X_4 are the children. These connections are shown in Figure 4.4(a). All the different connections between the variables imply different level of correlation. For example, there is a direct correlation between the two children X_3 and X_4 of T due to their common parent. Since X_1 and X_2 are ancestors of X_3 and X_4, there is also a big correlation between these variables. These correlations are reported by black lines on Figure 4.4(b). The dashed line between the two parent translates the fact that there might be a correlation between these two variables but it is not necessary. Indeed, X_1 and X_2 may be totally independent and not be correlated at all. This possible correlation between the parents of the target can be a problem through the whole process of this algorithm because the correlation present in the observational dataset may disappear in the experimental dataset if one of the parent is manipulated as shown in Figure 4.5(b) while conserving the correlation with the children. In Figure 4.5(a), we can see the effect of a manipulation on a child on the correlation between this child and the other members of the Markov Boundary of the target. All the correlations between a manipulated child and the other variables of the Markov Boundary are strongly affected and this algorithm attempts to highlight these variations to detect a manipulation. Indeed, an intervention on a parent only affects the hypothetical correlation with the other parents but the correlations with the unmanipulated children are conserved inducing small variations between the two datasets.



(a) Example of a causal graph (b) Correlation between the variables

Figure 4.4: Example of a local causal graph and the correlations between the variables without manipulation (a black line corresponds to a correlation and a dashed line represents a possible correlation)



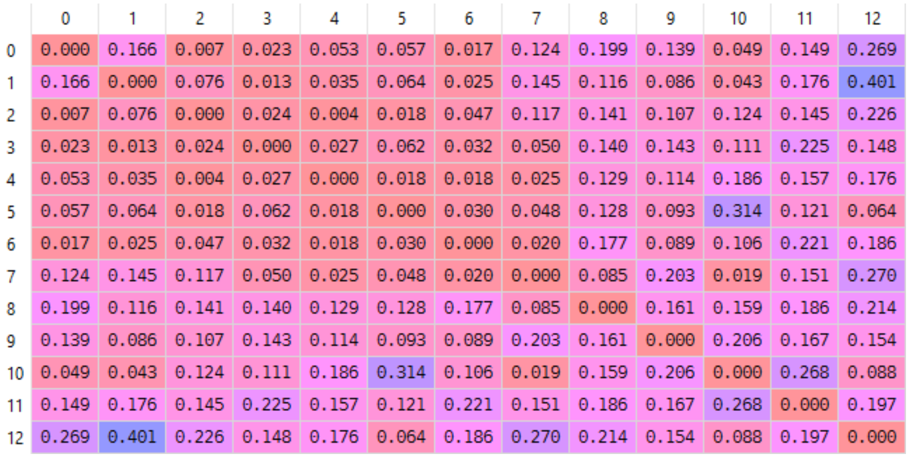
(a) Example of a manipulation on a child (b) Example of a manipulation on a parent

Figure 4.5: Effects of a manipulation on a parent or a child on the correlations between the variables in the Markov Boundary with a black line representing a correlation and a red line the absence of correlation

From these different possible cases, we can highlight three different areas in the variation of correlation between observational and experimental dataset :

- Big variations when a child is manipulated
- Small variations when a parent is manipulated because of the possible correlation between the parents in the observational dataset
- Very small variations when there are no manipulations

The two last areas include all the good features and must be kept. In figure 4.6, we can see the three different areas. The first area corresponds to the last six columns and lines. The second area can be seen in the two first lines and columns because they correspond to the parents. The high values of $K_{0,1}$ and $K_{1,0}$ are explained by the manipulation of one of the parent. The lines and columns from 2 to 6 correspond to the last area.



	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.000	0.166	0.007	0.023	0.053	0.057	0.017	0.124	0.199	0.139	0.049	0.149	0.269
1	0.166	0.000	0.076	0.013	0.035	0.064	0.025	0.145	0.116	0.086	0.043	0.176	0.401
2	0.007	0.076	0.000	0.024	0.004	0.018	0.047	0.117	0.141	0.107	0.124	0.145	0.226
3	0.023	0.013	0.024	0.000	0.027	0.062	0.032	0.050	0.140	0.143	0.111	0.225	0.148
4	0.053	0.035	0.004	0.027	0.000	0.018	0.018	0.025	0.129	0.114	0.186	0.157	0.176
5	0.057	0.064	0.018	0.062	0.018	0.000	0.030	0.048	0.128	0.093	0.314	0.121	0.064
6	0.017	0.025	0.047	0.032	0.018	0.030	0.000	0.020	0.177	0.089	0.106	0.221	0.186
7	0.124	0.145	0.117	0.050	0.025	0.048	0.020	0.000	0.085	0.203	0.019	0.151	0.270
8	0.199	0.116	0.141	0.140	0.129	0.128	0.177	0.085	0.000	0.161	0.159	0.186	0.214
9	0.139	0.086	0.107	0.143	0.114	0.093	0.089	0.203	0.161	0.000	0.206	0.167	0.154
10	0.049	0.043	0.124	0.111	0.186	0.314	0.106	0.019	0.159	0.206	0.000	0.268	0.088
11	0.149	0.176	0.145	0.225	0.157	0.121	0.221	0.151	0.186	0.167	0.268	0.000	0.197
12	0.269	0.401	0.226	0.148	0.176	0.064	0.186	0.270	0.214	0.154	0.088	0.197	0.000

Figure 4.6: Example of the K matrix for the DCED algorithm with $\mathcal{X}_{\text{hiton}}$

The implementation of this algorithm is basically the same as the DPED algorithm with a different function to compute the matrix M. The prediction and the error rate are replaced by a correlation coefficient and the difference between correlation coefficient for each feature. The matrix M becomes :

$$M_{i,j} = |C(X_i, X_j)| \tag{4.5}$$

With $C(X_i, X_j)$ a coefficient corresponding to the level of correlation between the variables X_i and X_j. The matrix K is computed with the same equation presented in the DPED section (see Equation 4.2).

The score of each feature is computed in the same way as the DPED algorithm by summing the columns in the matrix K. At this stage, we are expecting to see three distinct groups in the score array because of the three different areas discussed before. A "max increment" and "last" procedures to removed the bad features are also implemented based on the same idea developed for DPED.

4.3.3 Limitations

Like the DPED algorithm, the DCED algorithm takes an approximated Markov Boundary as input which can be partially wrong (some false positives and some irrelevant variables could be included in the approximated Markov Boundary). The score of the independent variables is very small since they are uncorrelated in both observational and experimental dataset (a manipulation on an independent variable do not significantly changes the correlation with the other variables). First, they are always included in the final subset but they also may push good features out of the final subset due to their really small scores. While false positives variables (variables considered as being in the Markov Boundary while they are not) do not behave like the parents and the children and so may lead to wrong conclusion about their nature (good vs bad). A pre-processing is really important to remove these features. An example of this limitation is showed in the results chapters.

When there are too many manipulations like in REGED2 where all the children are manipulated for example, the DCED algorithm cannot uncover the parents since all the correlations between the children and the parents are removed because of the interventions. Since the correlations between the parents in the observational data set are random, the variations of the correlation coefficient between observational and experimental dataset may be non zero. In that case, DCED will also reject the parents because there are also variations between observational and experimental data set and this effect cannot be counterbalanced by the low score between the parents and the unmanipulated children because all the children are manipulated in REGED2.

Chapter 5

Results based on the observational data set

This chapter is dedicated to a predictive approach only based on the observational dataset without directly taking into account the manipulated data. A comparison between different learning algorithms and their performances on both observational and experimental dataset is drawn before applying any feature selection. This approach does not take advantages of any particular underlying mechanism in the database to predict the target. Then, we introduced some feature selection and compared both classical and causal feature selection and their impact on the performances. In Section 5.1.2, we focused on the Random Forest family of algorithm to be able to compare the performances of the different feature selection methods.

We used the library Scikit Learn for all the learning algorithm.

5.1 Reged dataset

In this section, we will show the results obtained with the dataset REGED described before. Since it is a classification problem, we used a specific statistical test for filtering and some specific classifier algorithms like linear support vector classifier (LinearSVC) for the classical feature selection. For causal discovery, we focused on two algorithms : HITON PC and IAMB. They are quite efficient to discover local Markov Boundary while staying generic. Indeed, they are one of the basic tools for local causal discovery while other algorithms may be more specific to particular dataset. The IAMB algorithm can be used for both classification and regression but HITON PC can only be used for classification. For this reason, we applied both to the REGED dataset but only IAMB to the P1000 dataset. We first put in contrast classical and causal feature selection to discover the Markov Boundary followed by an analysis of the impact of the different feature selection on prediction.

5.1.1 Classical approach without feature selection

The principles of the classical approach is to learn a model on all the features of the observational dataset in order to predict the target of further test set. This approach is naive and could lead to very bad performances for some algorithms which need some preprocessing and a good set of parameters. The next step should be a cross validation over the parameters of all the learning algorithms but we decided to only focus on the preprocessing such as feature selection because we are interested in the tendencies but not in the absolute performances.

The classifiers that we considered in this section are :

- Random Forest [5]
- Extra Trees [7]
- Linear SVC [6]

We left all the parameters by default except for the number of estimators for the tree algorithms which we set at 1000.

	Random Forest	ExtraTrees	Linear SVC
Reged0	0.8164	0.6302	0.9695
Reged1	0.5114	0.5136	0.7545
Reged2	0.5011	0.5008	0.5703

Table 5.1: Scores of the predictions for REGED with different learning algorithms and all the features

In Table 5.1, we can see the scores for the different test sets with the different learning algorithms presented before. The SVC classifiers are quite good on REGED0 while tree ensemble classifiers with default parameters are much less efficient. However, the scores drastically decrease between the observational test set (REGED0) and the experimental test set (REGED1 and REGED2) for all classifiers. These differences can easily be explained by the variations of the distribution between the datasets. When the number of manipulated features increases, the score decreases as it can be seen in the two lines corresponding to REGED1 and REGED2 since the number of manipulations is larger in REGED2 than in REGED1.

5.1.2 Classical vs Causal feature selection for discovering the Markov Boundary

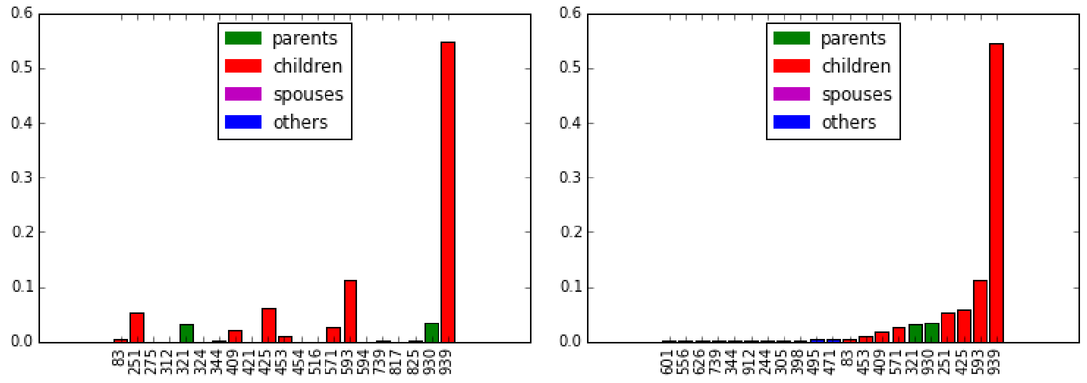
The preprocessing mentioned before mainly consists in selecting a good set of features on which the model is learned to improve the performance of the prediction. There are a lot of different feature selection techniques and their performances vary a lot from a situation to an other. This thesis do not pretend to assess all the techniques, therefore, we focused on three classical and two causal methods for the REGED database :

- Classical :
  - Random Forest *feature_importances_* attribute
  - Linear support vector absolute weight ranking
  - t-test filter
- Causal :
  - HITON pc [3]
  - IAMB [13]

One important reason of causal feature selection is the fact that the Markov Boundary of a target is supposed to be an optimal subset of features for prediction. Since the Markov Boundary is the smallest set of features containing all variables carrying information that cannot be obtained from other variables, this subset is only composed of strongly relevant and weakly relevant features. Some weakly relevant features may be redundant. For example, if two children have the same amount of information as a parent, they are redundant but are both in the Markov Blanket. Hence, retrieving the Markov Boundary is a crucial step for predicting a variable if the dataset is obtained from a causal distribution. Nevertheless, the Markov Boundary is not enough to predict a variable if the distributions are not the same between the learning and the test set (if there are some manipulations). We are expecting good results on the observational test set (REGED0) but poor performances on both experimental test set (REGED1 and REGED 2) with the Markov Boundary subset because we do not take into account the manipulations when we learn the model.

Since we have access to the local causal graph around the target for the REGED dataset, we perfectly know the unique Markov Boundary since we assumed the faithfulness between the graph and the distribution. We can thus compare both classical and causal feature selection with the true Markov Boundary. The local causal graph is described in Chapter 3. There are 21 features in the Markov Boundary : 83, 251, 275, 312, 321, 324, 344, 409, 421, 425, 453, 454, 516,571, 593, 594, 739, 817, 825, 930, 939 on which we focused our analysis.

The classical feature selection techniques are divided in two stages : ranking and selection. The first step is to compute a score for each variable depending on their importance or the information that they contain and the second step is to select the k best features as the new subset. The Markov Boundary being assumed to be an optimal subset, we are expecting the features present in the Markov Boundary to be the best features in the ranking of the classical techniques. We therefore used $k = 21$ to show the ability of classical methods to retrieve the Markov Boundary.



(a) Scores for the true Markov Boundary features

(b) Scores for the 21 more important features

Figure 5.1: Scores for the Markov Boundary features and the 21 more important features from RandomForest algorithm *feature_importances_* attribute (max_features = None, n_estimators = 1000 and random_state = 1) for the REGED dataset

The 21 best features found with a Random Forest ranking can be found in Figure 5.1(b). We called this subset $\mathcal{X}_{\text{best 21}}$. The very high score compared to the others of feature 939 translates the high information contained in this child about the target T. This score also implies that if there is a manipulation on this feature, the model will give a lot of importance to a disconnected variable. An intervention on a very lowly predictive variable is not so harmful if the set of features is big enough but when the subset is small a single manipulated feature can be very harmful. Indeed, when an irrelevant variable is manipulated in the whole set, the consequences are limited since most of the learning algorithm gives low importance to irrelevant variables. It is possible that manipulations on features included in the Markov Boundary lead to smaller impact than expected if the learning algorithm does not assign to much importance to them. This phenomena can explain the differences between the performances on REGED1 of the algorithms in Table 5.1.

From Figure 5.1(a), we can conclude that the distribution of the information among the Markov Boundary is clearly not identical but the two parents rank high while spouses do not fare well.

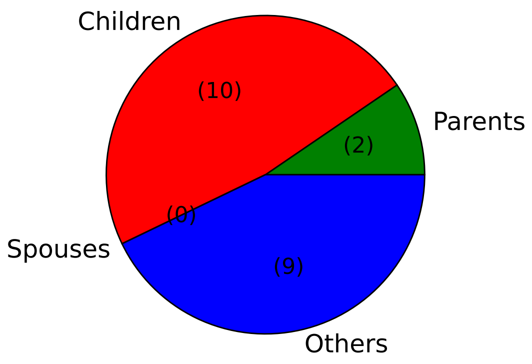


Figure 5.2: Nature of the 21 more important features computed with the *feature_importances_* attribute of the Random Forest Algorithm for the REGED dataset

In Figure 5.2, we can see the nature of the 21 best features based on the ranking of Random Forest algorithm. Although, we expected to find all the Markov Boundary, we found a lot of features not included in the unique Markov Boundary (we labeled them as *Others* in the pie charts). Their presence can be explained by the redundancies between variables. Indeed some highly predictive variables can share their information with other variables and even if only one variable could be enough to predict the target, the model will learn from all the variables. The explanation can also come from the violation of the faithfulness assumption. In conclusion, these *Others* features can be highly predictive even if they are not part of the set of parents, children and spouses.

We repeated this analysis with a linear support vector classifier with the default parameters. There were no major differences between the distribution of the 21 best features but we can see in Figure 5.3 that there are two more children than in the previous feature selection. Random Forest and Linear SVC are embedded methods and work on similar principles. These methods were unable to find spouses for this dataset.

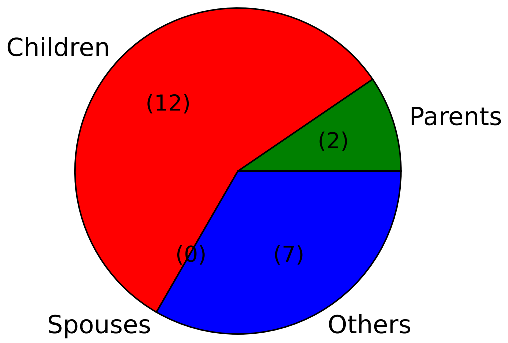
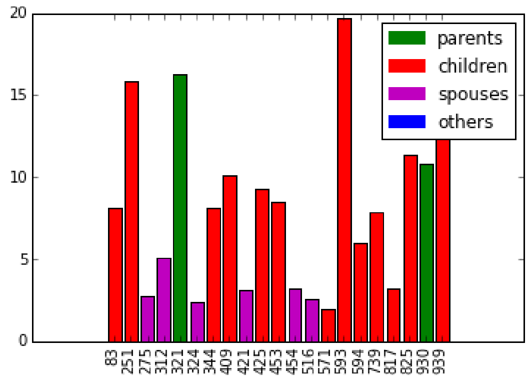
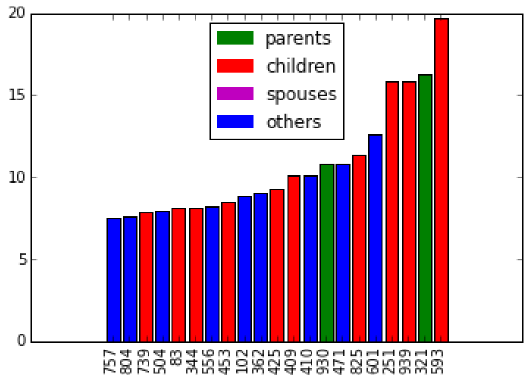


Figure 5.3: Nature of the 21 best features computed with the linear Support Vector Classifier absolute weight ranking for the REGED dataset

We also wanted to show the results with a filter method. We used the t-test filter for this purpose. This test computes the dependencies between each feature and the target giving an insight into the underlying mechanisms around the target. We are expecting a low correlation between the spouses and the target because they are not directly connected to it as it can be seen in Figure 5.4(a). Some features have a lower score than spouses showing the difficulties to retrieve the whole Markov Boundary since some directly connected features have lower correlation than spouses or others features (features labeled as *Others*). The overall distribution of the 21 best features is almost the same for all the classical techniques tested in this thesis and their nature is almost the same for the three methods (See Figure 5.2, 5.3 and 5.5).



(a) Scores for the Markov Boundary features



(b) Scores for the 21 more important features

Figure 5.4: Scores for the Markov Boundary features and for the 21 more important features from univariate filter with t-test for the REGED dataset

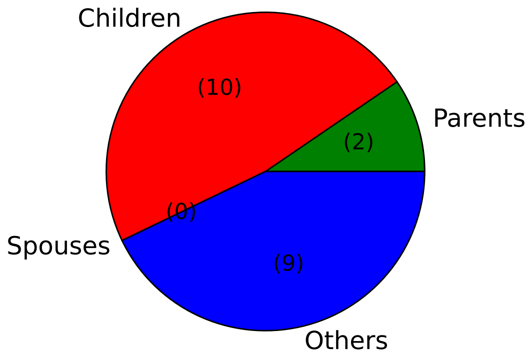


Figure 5.5: Nature of the 21 best features computed with the t-test filter for the REGED dataset

We assumed that the Markov Boundary was an optimal subset for predicting the target. Hence, we considered a feature labeled *Others* as a wrong feature since it is not a member of the Markov Boundary. However, to assess the efficiency of a subset, we need to predict on a test set and compute the score to do a rigorous comparison. In Table 5.2, we can see the scores for the different feature selection on the three test sets of REGED. The prediction were done with a Random Forest classifier. On REGED0, the true Markov Boundary is the best subset as expected but the three other subsets have really close performances with a variation of less than 1% around the maximum score. We see a small increase of the score when we used a feature selection for REGED1 but for REGED2 there is no significant improvement. Finally, classical methods are good on the observational test set (without manipulations) but have limited impact on the experimental test set even if there is a small improvement.

	All features	Markov Boundary	linear SVC	Random Forest	t-test Filter
Reged0	0.8164	0.9835	0.9798	0.9774	0.9806
Reged1	0.5114	0.5751	0.5780	0.5653	0.5700
Reged2	0.5011	0.5140	0.5198	0.5216	0.5204

Table 5.2: Performances of the Random Forest algorithm for REGED with different classical feature selection

To overpass the limitations met with the classical approach, we introduced a causal approach. We used two local causal discovery algorithms : HITON PC and IAMB. The objective of these methods is to find the Markov Boundary with a limited number of errors. From the classical methods, we could highlight the small importance of some Markov Boundary variables. Therefore, we are expecting the causal methods to miss some lowly correlated variables like spouses for example. We decided to use HITON PC over HITON MB because of the low correlation between the spouses and the target. Indeed, HITON MB in order to find the spouses also included a lot of wrong features in the final subset because it could not tell the difference between spouses and some highly wrong correlated variables. In Table 5.3, we can see the output subset with the HITON PC algorithm and the percentage of feature for each sort discovered. Since this algorithm does not look for spouses it is normal not to discover any. By focusing on parents and children, the algorithm could reach a really good performance and found all the parents and 85% of the children without any errors.

	Features selected	percentage discovered
Parents	321,930	100%
Children	83, 251, 344, 409, 425, 453, 593, 594, 739, 825, 939	85%
Spouses	-	0%
Others	-	-

Table 5.3: Results from the HITON PC algorithm with a statistical threshold of 0.05

If we want to include the spouses in the search, we need to use a different algorithm. The IAMB algorithm was the most efficient among the compatible methods. The addition of the spouses lead to slightly worse performance as it can be seen in Table 5.4 especially for the children discovery. Moreover, to discover two spouses, the algorithm also added one wrong feature.

	Features selected	percentage discovered
Parents	321,930	100%
Children	251, 409, 453, 593, 594, 825, 939	54%
Spouses	312,516	33%
Others	413	-

Table 5.4: Results from the IAMB algorithm with a statistical threshold of 0.0001

Table 5.3 and 5.4 present the best results for each algorithm. The statistical threshold α has a huge impact on the efficiency of the algorithms. In order to do an in-depth analysis of the impact of α on the efficiency we plotted the number of features in the output subset as a function of α for both algorithms. We divided the features in two categories :

- Good features : this category includes all the members of the Markov Boundary (parents, children and spouses)
- Wrong features : this category includes all the variables external to the Markov Boundary

At this stage, we used these definitions to categorize the features but later in the report we had to slightly change these definitions to take into account a new concept that we introduced in the next chapter.

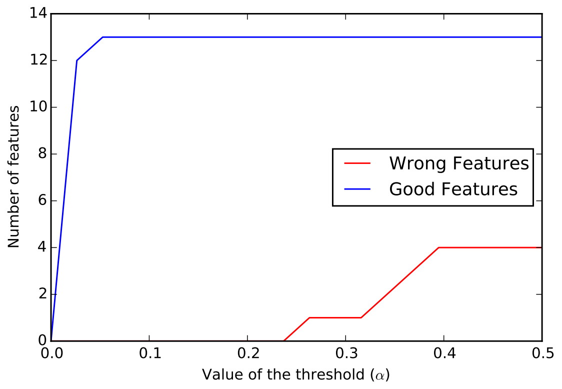
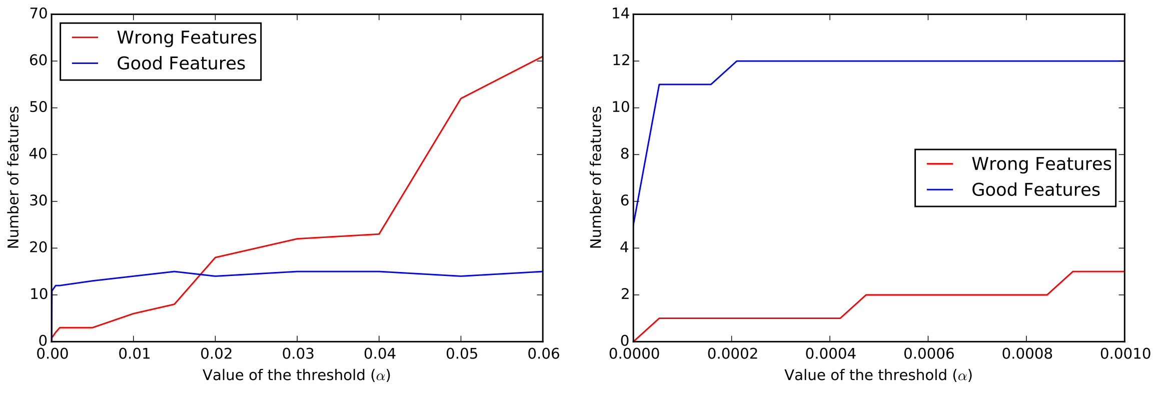


Figure 5.6: Features selected as a function of the statistical threshold with the HITON PC algorithm for the REGED dataset

Figure 5.6 shows the evolution of the selected features as a function of α. For a value of the parameter smaller than 0.24, there is no wrong feature in the output subset and between 0.05 and 0.24 we can see that the number of good features discovered is maximum. It means that any value in $[0.05, 0.24]$ gives the best result with the HITON PC algorithm for the REGED dataset. Another interesting point is that the greater α is the more features are included in the output subset and the error curve increases as well.



(a) Large variations of the threshold

(b) Zoom on the first part of the graph

Figure 5.7: Features selected as a function of the statistical threshold with IAMB algorithm for the REGED dataset

The important range of the parameter α is sensitively different between the two algorithms. While the good range for HITON PC is $[0.05, 0.24]$, the good range for IAMB is $[0.00010.0004]$. There is a factor 100 between the two ranges because of the inner differences between the two algorithms despite the same default value for α of 0.05 provided by the Causal Explorer library. Around the default value, the IAMB algorithm includes a lot of wrong features as it can be seen in Figure 5.7(a) but has the same kind of behavior in its good range as the HITON PC algorithm (see Figure 5.6 and 5.7(b)).

The subset found with the algorithm HITON PC with $\alpha = 0.05$ is called $\mathcal{X}_{hiton}$ and the subset found with the algorithm IAMB with $\alpha = 0.0001$ is called $\mathcal{X}_{IAMB}$. Their scores for the prediction on the three test sets are reported in the Table 5.5. The results are almost the same as with classical features selection but the score with $\mathcal{X}_{hiton}$ is a little bit better on REGED0 than the score with $\mathcal{X}_{IAMB}$.

	HITON PC	IAMB
Reged0	0.9850	0.9740
Reged1	0.5789	0.5749
Reged2	0.5150	0.5266

Table 5.5: Scores of the predictions for REGED with two different causal feature selection (HITON PC with $\alpha = 0.05$ and IAMB with $\alpha = 0.0001$) obtained with a Random Forest learning algorithm

We used the output subset $\mathcal{X}_{hiton}$ from the HITON PC algorithm with a statistical threshold of 0.05 reported in the Table 5.3 for the next chapter. A deeper analysis of the features selected is necessary to fully understand the further steps to the final subset for both REGED1 and REGED2 described in the next chapter.

	unmanipulated	manipulated
Parents	930	321
Children	344, 425, 453, 594, 739, 825	83, 251, 409, 593, 939
Spouses	-	-
Others	-	-

Table 5.6: Overview of the manipulations in the subset of features selected by the algorithm HITON PC with the statistical threshold equals to 0.05 for the REGED1 database

At the end of this section, we selected a subset of features as an approximated Markov Boundary. In this subset, there are potentially wrong features (features which do not belong to the Markov Blanket but still selected due to errors in the causal discovery algorithm) and bad features for prediction (manipulated features except the parents). The local graph (both datasets) and the list of manipulated features (for REGED2) were not available during the challenge but we used them here to have a better understanding of the different steps of the feature selection process. Since we had access to the local graph and to the list of manipulated features, we knew which features we had to remove in order to have a subset with only predictive features for the experimental dataset.

In the Table 5.6, we can see the distribution of manipulations in the subset selected by the HITON PC algorithm. The features we tried to remove for the REGED1 dataset are : 83, 251, 409, 593, 939. Indeed, the selected subset was composed of only parents and children so only the manipulated children are independent of the target and not predictive anymore. In the case of REGED2, all the children are manipulated so the only good features are the parents : 321 and 930.

5.2 P1000 dataset

We wanted to test our methodology on other datasets. The P1000 dataset and the REGED dataset differ by their target nature. While the REGED target variable is discrete, the P1000 target variable is continuous. This difference is very important because the learning algorithms are different even if they come from the same family of algorithms. The problem proposed by the P1000 database is a regression problem and not a classification. We followed the same structure as in the previous section.

5.2.1 Classical approach

The regressors that we considered in this section are :

- Random Forest regressor
- Extra Trees regressor
- Linear SVR
- SVR with a rbf kernel

Since we only want to highlight the tendencies and not the absolute value of the score, we left all the parameters by default except the number of estimators for the tree algorithms to have significant results. We reported all the results in the Table 5.7. We used the mean absolute error (see 3.2).

	Random Forest	ExtraTrees	Linear SVR	SVR with rbf kernel
P1000 observational	0.1103	0.107	0.1131	0.1526
P1000 experimental	0.7327	0.6655	0.6125	0.2863

Table 5.7: Scores of the predictions for P1000 with different learning algorithms and all the features

The two algorithms based on tree ensemble and the two support vector regressors have similar behavior. Although, the Random Forest and Extra Trees algorithms seem to have better performance on the observational test set, the SVR family have sensitively better performance on the experimental test set. This difference comes from the importance given to each feature by the two families. As we will show in Section 5.2.2, it seems that tree ensemble family focuses the learning on the children while support vector family focuses on a more mixed set composed of parents and children. This difference has low effects on the performance for the observational test set but significant effects for the experimental test set. Indeed, if an algorithm gives much importance to a children which is manipulated in the test set, it leads to poor performance on the test set but it is not the case if the algorithm gives much importance to a parent because even manipulated it is still predictive.

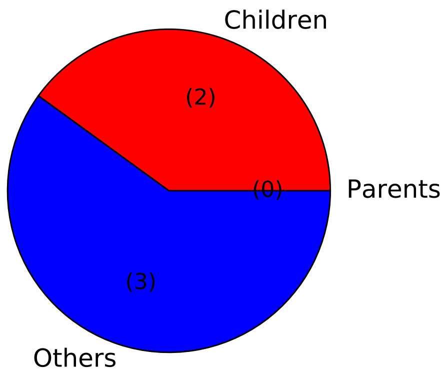
5.2.2 Classical vs Causal feature selection for discovering the Markov Boundary

The P1000 dataset differs from the REGED dataset because we strictly know the whole graph since this set is entirely artificial while for REGED we only knew the local graph around the target. We perfectly know the Markov Boundary but also the features around the Markov Boundary. As for the REGED dataset, we focused our analysis on the the size of the Markov Boundary. Here we considered the five best features.

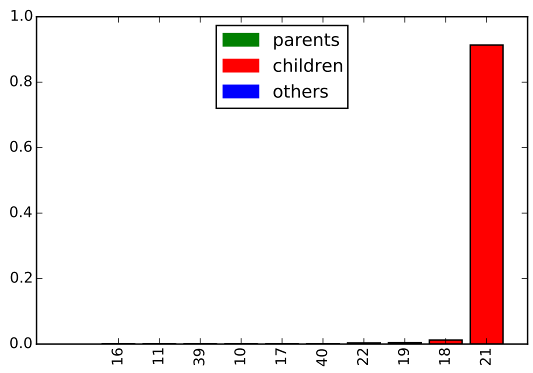
The feature selection techniques used for regression are slightly different from those used for classification. We still used rankings from Random Forest and SVM families but we had to replace the t-test ranking by the Pearson ranking to compute the correlation coefficients between the target and the features for classical methods. As said previously, the algorithm HITON PC is not adapted for regression. Hence, we only focused on IAMB with different ranges of values for the parameter α. Here is a list of the techniques used in this section :

- Classical :
  - Random Forest features importances attribute
  - Linear support vector absolute weight ranking
  - Pearson ranking
- Causal :
  - IAMB with two distinct ranges of α

As said previously, the importance given to the children by the random forest algorithm is much higher than the importance given to the parents. Indeed, as it can be seen in Figure 5.8(b), one children has a score higher than 90% and it is almost the only useful feature for this learning algorithm. The nature of the five best features is shown in Figure 5.8(a). We can notice that there is no parent in the top five features which explains the bad performance on the experimental set shown in Table 5.8.



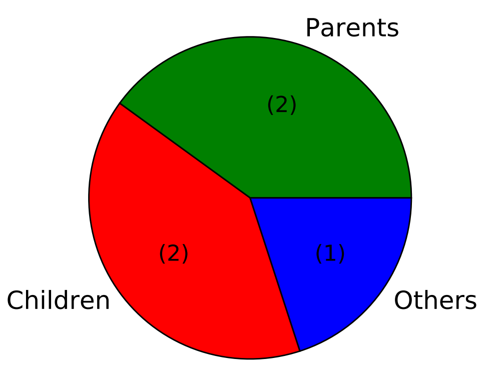
(a) Nature of the 5 more important features



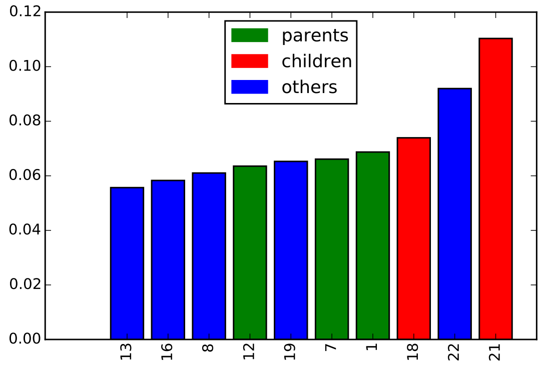
(b) Scores for the 10 more important features

Figure 5.8: Nature of the 5 more important features and scores for the 10 more important features from Random Forest algorithm features importance attributes (n_estimators = 1000 and random_state = 1) for the P1000 dataset

The SVR algorithm has a more mixed result even if the best feature is still a child (see Figure 5.9(b)). There are two parents and two children in the top five features. This a more normal result than the subset obtained with the Random Forest ranking since we are expecting the parents to have a big score too. If we analyze a little bit deeper the feature labeled as *Others*, we realize that it is a redundant feature of a child excepting the noise. From this observation, it is normal to find it in the top five features.



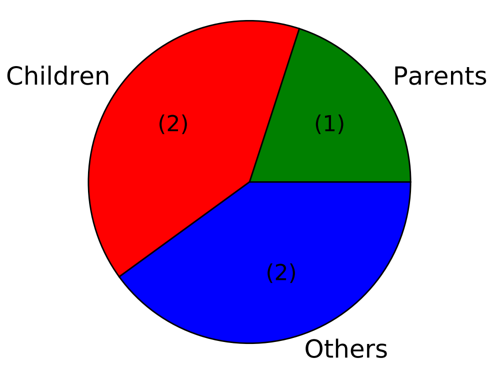
(a) Nature of the 5 best features



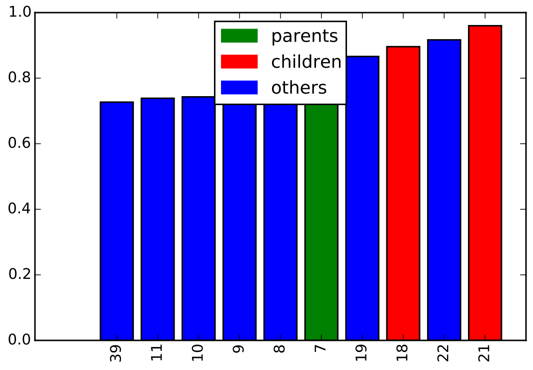
(b) Scores for the 10 more important features

Figure 5.9: Nature of the 5 best features and scores for the 10 more important features from Support Vector Regression with a linear kernel for the P1000 dataset

We also computed the Pearson correlation coefficient between each feature and the target to build a ranking that can be used as a filter. This approach leads to a mixed ranking with more *Others* features as it can be seen in Figure 5.10(a).



(a) Nature of the 5 best features



(b) Scores for the 10 more important features

Figure 5.10: Nature of the 5 best features and scores for the 10 more important features from univariate filter with Pearson coefficient for the P1000 dataset

The Markov Boundary subset has the best performance on the observational test set but really poor performance on the experimental test set because all the children are manipulated (see Table 5.8). We can also notice that all the subsets including children have very bad performance on the experimental test set while keeping good results on the observational test set. Some feature selection techniques may give more importance to the parents than the children. In that case, the results on the experimental test set could be sensitively better.

	All features	Markov Boundary	SVR	Random Forest	Pearson coefficient
P1000 observational	0.1103	0.0960	0.1039	0.1132	0.1117
P1000 experimental	0.7327	0.8033	0.8308	0.8781	0.8524

Table 5.8: Scores of the predictions made with a Random Forest Regressor for P1000 with different classical feature selection (For SVR, Random Forest and Pearson coefficient, the score is computed with the 5 more important features)

There is a big disparity in the nature of the features in the subsets used to compute the scores of the Table 5.8. Moreover, we are not supposed to know the size of the Markov Boundary and so the best value of k (the number of selected features). We could do a cross validation to find the best number of features for each algorithm but this is an expensive approach and the results may never converge to a good solution. Indeed, if the first feature is a child, there will be always a child in the subset and if this child is manipulated all the score may be bad whatever the value of k.

We highlighted before that the Markov Boundary is the optimal subset for the observational dataset but classical techniques used to retrieve it are not robust and there are variations between the different methods. A more robust way to find the Markov Boundary and so an optimal set of features for the observational dataset is to use a local causal discovery algorithm. The IAMB algorithm was the most appropriate method for a regression problem. We were able to plot the evolution of the number of good and wrong features in the output subset $\mathcal{X}_{IAMB}$ as a function of the statistical threshold α because we had access to the local graph. Even for very low values of α, the algorithm find all the five members of the Markov Boundary but start to include wrong features very fast (see Figure 5.11).

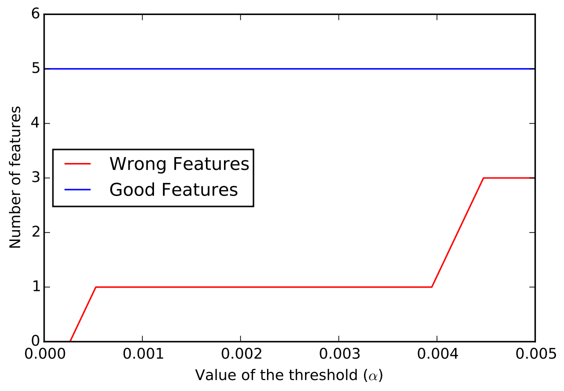


Figure 5.11: Number of selected features as a function of the statistical threshold with the IAMB algorithm for the P1000 dataset

In order to stay generic, we used two distinct values of α to generate two approximated Markov Boundary : $\mathcal{X}_{IAMB1}$ and $\mathcal{X}_{IAMB2}$. The details of these subsets are found in Table 5.9. These subsets are used in the next chapter. We can see that $\mathcal{X}_{IAMB1}$ is a perfect approximation of the Markov Boundary while $\mathcal{X}_{IAMB2}$ includes three additional wrong features.

	$\mathcal{X}_{IAMB1}$ with $\alpha = 0.0002$		$\mathcal{X}_{IAMB2}$ with $\alpha = 0.005$	
	Features selected	percentage discovered	Features selected	percentage discovered
Parents	1,7,12	100%	1,7,12	100%
Children	18,21	100%	18,21	100%
Spouses	-	-	-	-
Others	-	-	314,578,748	-

Table 5.9: Overview of the subsets computed with the IAMB algorithm with a statistical threshold α of 0.0002 and 0.005 for the P1000 database

It is important to keep in mind the distribution of the manipulations inside the output subset because even if the IAMB algorithm with good value of α can find the Markov Boundary, it does not mean that the predictions will be good on an experimental test set (as shown in Table 5.8). We still need to remove the non-parent manipulated features. These manipulations for $\mathcal{X}_{IAMB1}$ are reported in the Table 5.10.

	unmanipulated	manipulated
Parents	7,12	1
Children	-	18,21
Spouses	-	-
Others	-	-

Table 5.10: Overview of the manipulations in the subset of features selected by the algorithm IAMB with $\alpha = 0.0002$ for the P1000 database

The performances of $\mathcal{X}_{IAMB1}$ and $\mathcal{X}_{IAMB2}$ are shown in the Table 5.11. The scores on the experimental test set are very bad but $\mathcal{X}_{IAMB1}$ has a slightly better performance than $\mathcal{X}_{IAMB2}$. In order to improve their performances on the experimental test set, we need to remove the non-parent manipulated features .

	$\mathcal{X}_{IAMB1}$	$\mathcal{X}_{IAMB2}$
Observational	0.0960	0.1011
Experimental	0.8033	0.8096

Table 5.11: Prediction scores for P1000 with the subsets computed with the IAMB algorithm with a statistical threshold α of 0.0002 and 0.005

5.3 Conclusion

The classical feature selection have big impact on the performance for the observational test set. The causal feature selection techniques have slightly better performance than classical ones on the observational test set. On REGED, the performances are slightly better with the causal feature selection on the manipulated data but on P1000, the performances are severely degraded because the models focused on manipulated children.

In the next chapter, we will try to find and remove the non-parent manipulated features and the irrelevant features in order to improve the predictions on the experimental test set for both dataset.

Chapter 6

Results using the manipulation information

6.1 Introduction

The DPED and DCED algorithms are quite sensitive to the input subset of features. These algorithms have really bad performance if they are used with all features. In order to provide a suitable subset, some preprocessing must be done. One approach is to approximate the Markov Boundary because this subset is enough to predict the target and all the features are correlated with the target. The two algorithms used in this thesis are HITON PC and IAMB because they are the most representative of the current causal discovery algorithm used through the world.

The next step is to remove all the lowly predictive variables included in the selected features from the two causal discovery algorithms. There are different ways to achieve a good sorting of the features. We mainly explored classical feature selection such as filter or embedded methods.

At this stage, the set of features is supposed to include only high predictive members of the Markov Boundary in the observational dataset. Some manipulated children and spouses are still included in the subset and we want to remove them. The DPED and DCED algorithms attempt to find them. At the end of these three steps, we have a subset of high predictive features in the experimental dataset.

Finally, we used this subset to predict the target of the experimental dataset. The different results are shown in the respective section in this chapter for both REGED and P1000 dataset.

Another approach is to redefine the notion of good and bad features. Previously, we considered a good feature as a member of the Markov Boundary still predictive in the experimental set but we do not need to restrict the research to the Markov Boundary since we are looking to improve the performances on the manipulated test set. If the purpose of the process is to predict the target without paying attention to the causal mechanisms, we can include all the unmanipulated features (even the features which are not member of the Markov Boundary) and the parents in the final subset (manipulated or not) in the good features. The new definitions of good and bad features are provided below :

- Good feature : A good feature is a highly predictive feature for the manipulated test set including the parents (manipulated or not) and all the unmanipulated features
- Bad feature : A bad feature is a lowly predictive feature for the manipulated test set including all the manipulated features except the parents

We used these definitions in this chapter for both approach even for the first one because an error (false positive) occurring during the causal discovery may still be a good feature. The difficulties is to define highly and lowly predictive, this is done with the help of the parameter k when we filter the subsets.

6.2 REGED

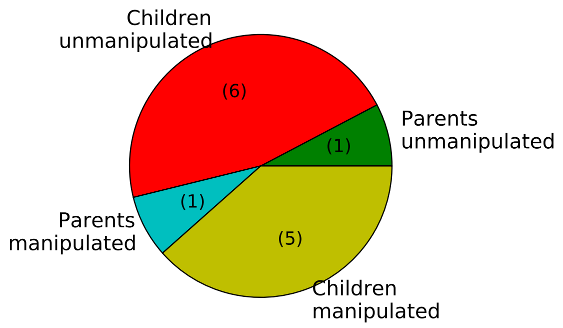
Before applying the DPED and DCED algorithms, we need to perform some filtering to remove the lowly predictive variables to avoid the problems that they generate (see Sections 4.2.3 and 4.3.3). We also show that this step is quite good to compensate the possible errors of the causal discovery algorithm. These errors are features not in the true Markov Boundary but still included in the approximated Markov Boundary (We labeled them as Other in the following sections). It is important to remove these features to discover the local causal graph but it is not so important for the final prediction if they are predictive and not manipulated.

Then, we will introduce the algorithms developed in the framework of this thesis to exploit manipulations in the test set.

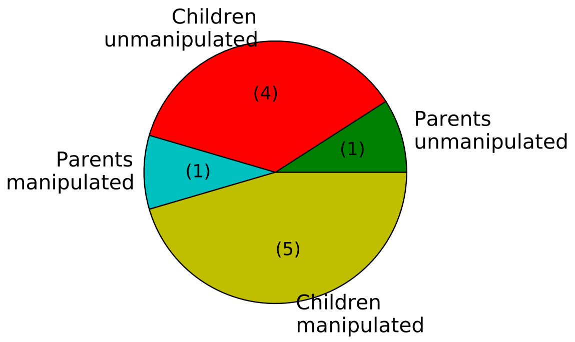
6.2.1 Feature selection in the approximated Markov Blanket

The purpose of this section is to identify and remove lowly predictive features. These features can be either wrong features (not in the true Markov Boundary) or some lowly predictive members of the Markov Boundary (manipulated or not).

The function used to do that is implemented in the DPED and DCED program under the name : *feature_filtering*. It compares a subset of features with the **k** best features ranked with the *feature_importances_* attribute of the Random Forest algorithm or with the *coef_* attributes if the provided model is a linear supervised vector machine and return only the features from the input subset that are in the **k** best features. The goal of this function is to eliminate the lowly predictive features to avoid miscellaneous problem while running both DCED and DPED algorithm.



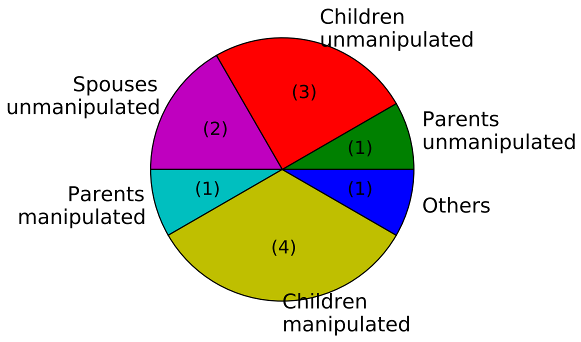
(a) Nature of the features in $\mathcal{X}_{hiton}$ with manipulations labels



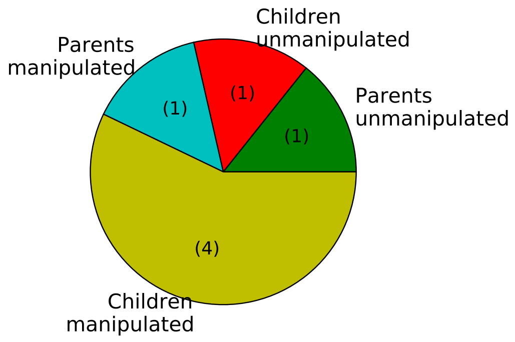
(b) Nature of the features in $\mathcal{X}_{hiton}$ with manipulations labels after filtering

Figure 6.1: Nature of the features of the subset $\mathcal{X}_{hiton}$ before and after filtering with the **20** best features ranked with the *feature_importances_* attribute of a Random Forest model (max_features = None, n_estimators = 1000) learned on all the features from the observational dataset for REGED1

From the Table 5.4, we can see that the feature 413 is not part of the true Markov Boundary and should be removed if it is a bad feature. As shown in the pie chart 6.2(b), the feature 413 is removed but also the two spouses previously discovered by the IAMB algorithm. Although these spouses are part of the true Markov Boundary, it seems that they are lowly predictive in this context. In the case of HITON PC, there is two removed features meaning that there are some lowly predictive children since both parents are kept as it can be seen in Figure 6.1(a) and 6.1(b) if we take $k = 20$. The concerned features are 594 and 825 which are direct children of the target. There could still be a predictive feature in the subset which is not part of the Markov Boundary. This category of features include both manipulated and unmanipulated features. The next step attempts to remove the manipulated ones. We called the subsets obtained after filtering $\mathcal{X}_{\text{hiton filtered}}$ for $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ for $\mathcal{X}_{\text{IAMB}}$.



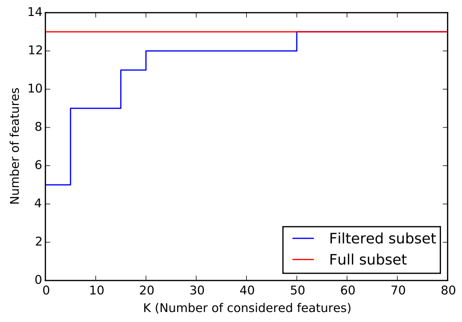
(a) Nature of the features in $\mathcal{X}_{IAMB}$ with manipulations labels



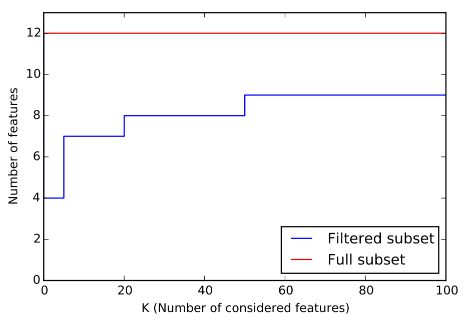
(b) Nature of the features in $\mathcal{X}_{IAMB}$ with manipulations labels after filtering

Figure 6.2: Nature of the features of the subset $\mathcal{X}_{IAMB}$ before and after filtering with the **20** best features ranked with the *feature_importances_* attribute of a Random Forest model (max_features = None, n_estimators = 1000) learned on all the features from the observational dataset for REGED1

We arbitrarily choose the value of k for the previous test but the choice of this value has a big impact on the next stages of the process. It is hard to quantify lowly and highly predictive. Therefore, a cross validation must be done to find the best value of k. The problem with a cross validation comes from the separation between the filtering and the DCED or DPED algorithm. The search of the best value of k must be conducted at the same time as the optimization of the DPED or DCED algorithm. We plotted the evolution of the number of features selected as a function of the parameter k for both $\mathcal{X}_{hiton}$ and $\mathcal{X}_{IAMB}$ in the Figure 6.3.



(a) With the $\mathcal{X}_{hiton}$ subset as input of the filter



(b) With the $\mathcal{X}_{IAMB}$ subset as input of the filter

Figure 6.3: Evolution of the number of features after filtering as a function of the **k** best features ranked with the *feature_importances_* attribute of a Random Forest model (max_features = None, n_estimators = 1000) learned on all the features from the observational dataset

6.2.2 DPED and results

In this section, we will present the results on REGED using the DPED algorithm. At this stage, we have different subsets of features obtained in the previous sections. We focused on five of them :

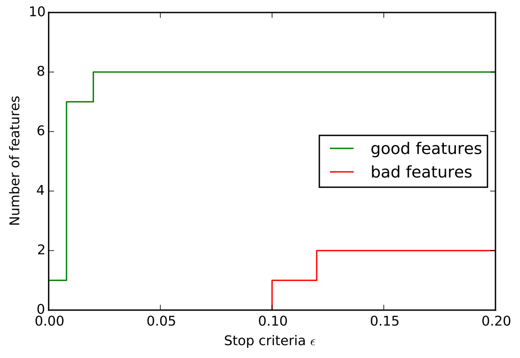
- $\mathcal{X}_{\text{hiton}}$: subset obtain from HITON PC algorithm with $\alpha = 0.05$
- $\mathcal{X}_{\text{IAMB}}$: subset obtain from IAMB algorithm with $\alpha = 0.0001$
- $\mathcal{X}_{\text{hiton filtered}}$: subset obtain after filtering $\mathcal{X}_{\text{hiton}}$
- $\mathcal{X}_{\text{IAMB filtered}}$: subset obtain after filtering $\mathcal{X}_{\text{IAMB}}$
- $\mathcal{X}_{\text{best 21}}$: subset containing the 21 best features from a Random Forest ranking

We first applied DPED to REGED1 and then to REGED2 and we analyzed the impact of the parameter ϵ on the output subset $\mathcal{X}_{DPED}$. We used the definitions of good and bad features explained in the introduction of this chapter. All the scores are computed with a Random Forest Classifier from Scikit Learn with 1000 estimators and the other parameters are left by default. The algorithm provided to DPED to predict all the inner models is also a Random Forest Classifier with 1000 estimators.

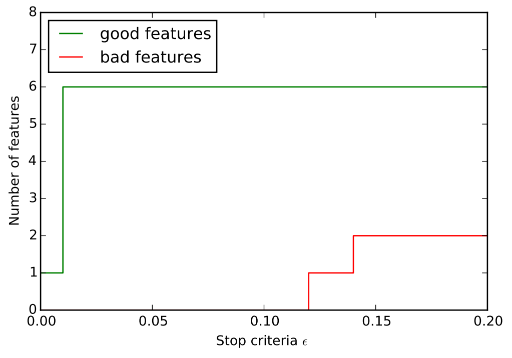
6.2.2.1 Reged1

The most important parameter of DPED is the threshold ϵ. Indeed, the higher ϵ is, the larger the number of features in $\mathcal{X}_{DPED}$ is. The problem is to find the optimal value of ϵ which maximize the performance of the prediction. The "last" mode was only efficient on $\mathcal{X}_{\text{best 21}}$ therefore we only showed the results of the max increment mode for the four other subsets.

In Figure 6.4, we can see that both $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ have the same kind of behavior but the maximum number of good features for $\mathcal{X}_{\text{hiton filtered}}$ is a little bit smaller than for $\mathcal{X}_{\text{hiton}}$. Here, the filtering was not particularly important even if it delayed the first appearance of bad features. We also could take a higher value of k to have $\mathcal{X}_{\text{hiton}} = \mathcal{X}_{\text{hiton filtered}}$. Under a particular value of ϵ which we called the critical ϵ (ϵ_{crit}), the output is only composed of one feature. In the case of $\mathcal{X}_{\text{hiton filtered}}$, after that particular value, we immediately got all the possible good features. One can clearly see the different distinct areas in both Figure 6.4(a) and 6.4(b). The first area is almost the same and appears for $\epsilon < 0.01$. For $\mathcal{X}_{\text{hiton filtered}}$, the second area spans over 0.01 to 0.12 and includes six good features while for $\mathcal{X}_{\text{hiton}}$ this area is divided in two parts and stops at 0.1 before including some bad features.



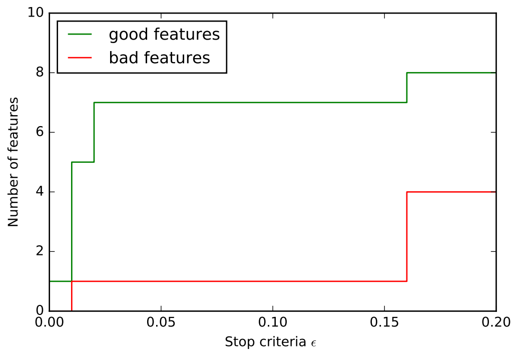
(a) With $\mathcal{X}_{\text{hiton}}$ as input



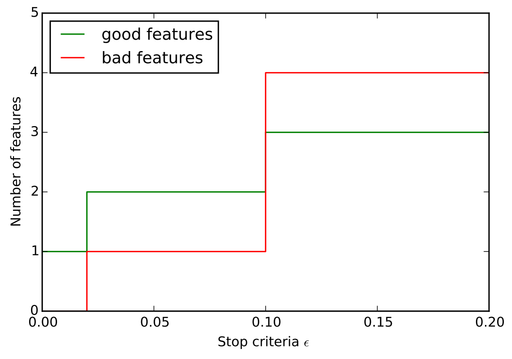
(b) With $\mathcal{X}_{\text{hiton filtered}}$ as input

Figure 6.4: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ as input of the DPED algorithm with the max increment mode for REGED1

The evolution of the composition of $\mathcal{X}_{DPED}$ with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ as input is shown in Figure 6.5(a) and 6.5(b). Unlike the subsets obtained with the algorithm HITON PC, the subsets $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ include directly after the critical ϵ a bad feature. We can notice that the filtering on $\mathcal{X}_{\text{IAMB}}$ did not have the intended effect. We were expecting it to remove the first bad feature included but this feature was not a lowly predictive variable. The bad feature concerned is the feature 409 which is a manipulated children. Throughout this section, the feature 409 was hard to process because its predictions on both observational and experimental datasets were quite bad and, therefore, it was hard to highlight the differences of predictions between the two datasets.



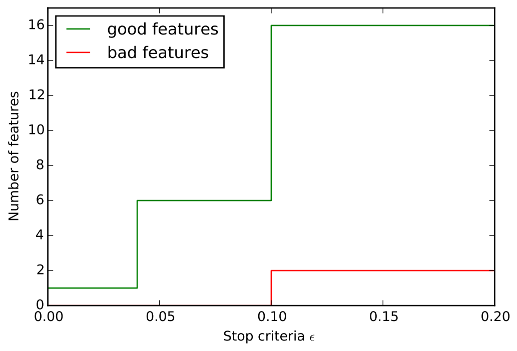
(a) With $\mathcal{X}_{\text{IAMB}}$ as input



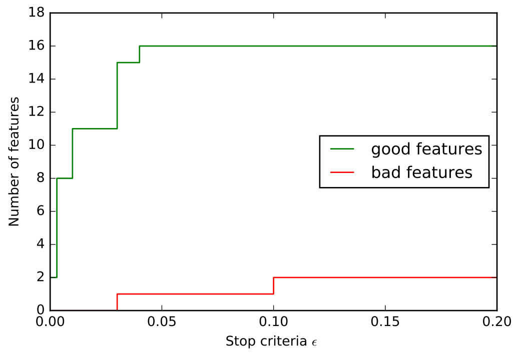
(b) With $\mathcal{X}_{\text{IAMB filtered}}$ as input

Figure 6.5: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ as input of the DPED algorithm with the max increment mode for REGED1

Since the DPED algorithm also works with a set of highly predictive features, we used $\mathcal{X}_{\text{best 21}}$ to show the differences between the two kind of preprocessing (causal and classical). On one hand, With the max increment mode, there are three distinct areas : under ϵ_{crit}, between 0.04 and 0.1 and one last area beyond 0.1 (see Figure 6.6(a)). On the other hand, with the last mode, the number of features in $\mathcal{X}_{DPED}$ increases on a more regular basis (see Figure 6.6(b)). Moreover, the last mode is able to include more good feature than the max increment mode before including a bad feature. We can see in Table 6.1, the score obtained with different values of ϵ and for the two different modes. The last mode has a significantly better performance for low values of ϵ and close performances for higher values. In this context, the last mode is really helpful because there are a lot of *Others* variables which have particular behavior and tends to make harder the search of the good features.



(a) With the max increment mode



(b) With the last mode

Figure 6.6: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with the 21 best features computed with the Random Forest *feature_ importances_* attribute as input of the DPED algorithm with the max increment and the last mode for REGED1

$\mathcal{X}_{\text{best 21}}$	$\epsilon = 0.01$	$\epsilon = 0.05$
max increment	0.6356	0.8180
last	0.8973	0.7831

Table 6.1: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) learned on a subset obtained by applying the DPED algorithm with the max increment and last mode and different values of ϵ to $\mathcal{X}_{\text{best 21}}$

The Table 6.2 shows the results for the prediction with $\mathcal{X}_{DPED}$ for different values of ϵ and different input subsets with the max increment mode. When $\epsilon = 0.05$, the subsets from HITON PC ($\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$) have the higher scores while for $\epsilon = 0.15$, $\mathcal{X}_{\text{IAMB}}$ has the best score because this is the only subset with only one bad feature for this particular value of ϵ.

	without DPED	$\epsilon = 0.05$	$\epsilon = 0.15$
$\mathcal{X}_{\text{hiton}}$	0.5789	0.9345	0.7647
$\mathcal{X}_{\text{IAMB}}$	0.5749	0.8108	0.8108
$\mathcal{X}_{\text{hiton filtered}}$	0.5706	0.9228	0.7373
$\mathcal{X}_{\text{IAMB filtered}}$	0.5609	0.7746	0.5609
$\mathcal{X}_{\text{best 21}}$	0.5653	0.8180	0.6813

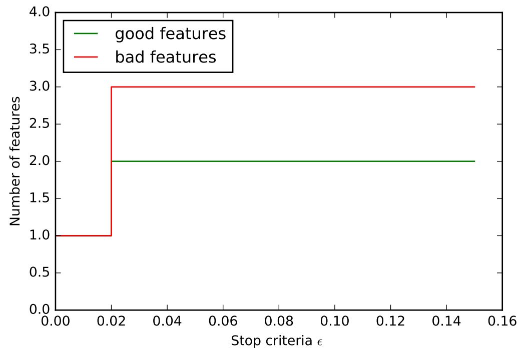
Table 6.2: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) for different subsets obtained by applying the DPED algorithm with the max increment mode and different values of ϵ to five different subsets

On overall, all the scores significantly increased from predictions without DPED and after processing with DPED if ϵ is small enough to avoid to include all the bad features (e.g. $\mathcal{X}_{\text{IAMB filtered}}$ with $\epsilon = 0.15$). The best score is obtained with $\mathcal{X}_{\text{hiton}}$ showing the importance of a causal feature selection before applying DPED. It is important to notice that a single bad feature in $\mathcal{X}_{DPED}$ drastically decreases the performance while the addition of a good feature does not always lead to an improve of the prediction.

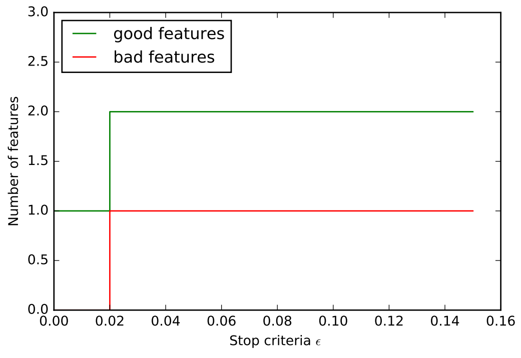
6.2.2.2 Reged2

After the good results obtained in the last section, we applied the DPED algorithm to an heavily manipulated dataset. We were expecting poorer performances on REGED2 than on REGED1 because only two variables of the unique Markov Boundary are still predictive and it is really hard to find them among the 999 features even with all the different feature selection applied on the initial subset. The good features are only the two parents while all the other features are considered as bad features since we do not know which feature is manipulated except the 13 children. Our set of bad features is not correct because we also considered as bad feature all the unamnipulated features not included in the unique Markov Boundary such as independent variables and parents of the parents of T. This can explained why we could have fair score with subset containing bad features. When taking the approximation on the bad features definition into account, the more important is to notice when DPED can find the parents and at which cost (the number of errors).

The utility of filtering can be seen in Figures 6.7(a) and 6.7(b). Indeed, the filtering removed lowly predictive features that would have been included in $\mathcal{X}_{\text{DPED}}$. The main difference between $\mathcal{X}_{hiton}$ and $\mathcal{X}_{\text{hiton filtered}}$ is the number of bad features because they both can find the two parents with the same value of ϵ. For $\epsilon < 0.02$ with $\mathcal{X}_{\text{hiton filtered}}$, the only feature is a parent while with $\mathcal{X}_{\text{hiton}}$ the two first feature are one bad feature and one good feature (here two features have the same score so they are both taken).



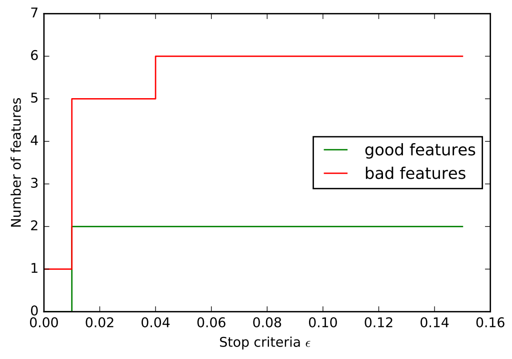
(a) With $\mathcal{X}_{\text{hiton}}$ as input



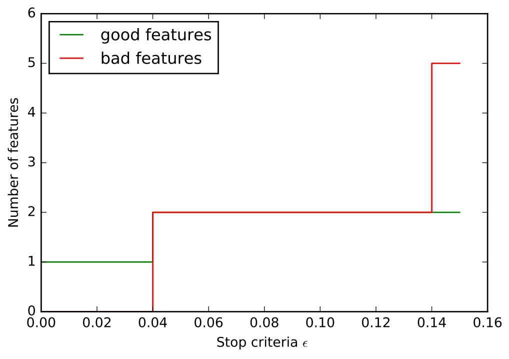
(b) With $\mathcal{X}_{\text{hiton filtered}}$ as input

Figure 6.7: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ as input of the DPED algorithm with the max increment mode for REGED2

The impact of filtering can also be noticed in Figures 6.8(a) and 6.8(b). Filtering has the same kind of effect on $\mathcal{X}_{\text{IAMB}}$ than on $\mathcal{X}_{\text{hiton}}$ even if the number of bad features is slightly higher with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$. They both manage to find the two parents.



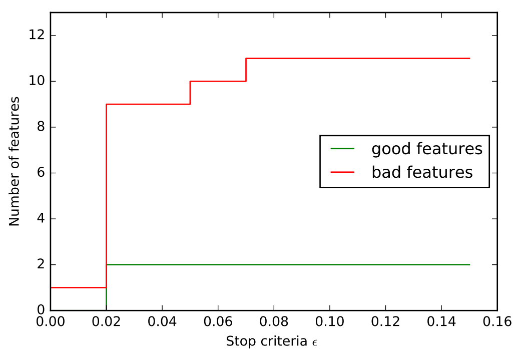
(a) With $\mathcal{X}_{\text{IAMB}}$ as input



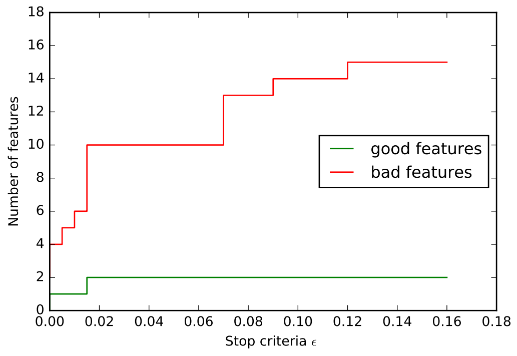
(b) With $\mathcal{X}_{\text{IAMB filtered}}$ as input

Figure 6.8: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ as input of the DPED algorithm with the max increment mode for REGED2

In Figure 6.9, we can see that both last mode and max increment mode included quite a lot of bad features and quite fast but most of the bad features in $\mathcal{X}_{\text{DPED}}$ subset obtained from $\mathcal{X}_{\text{best 21}}$ are mainly false negative. It means that we labeled them as bad features although they are still predictive. These false negatives features explained the high score obtained with $\epsilon = 0.04$ and $\mathcal{X}_{\text{best 21}}$ shown in Table 6.3.



(a) With the max increment mode



(b) With max increment mode

Figure 6.9: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with the 21 best features computed with the Random Forest *feature_ importances_* attribute as input of the DPED algorithm with the max increment and last mode for REGED2

The performances on this dataset are far less good than on REGED1 but we still can highlight some quite good results in particularly with $\mathcal{X}_{\text{hiton filtered}}$ and $\epsilon = 0.04$. The increase of the score between $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ shows the interest of filtering. Both subsets from IAMB algorithm did not predict very well even if $\mathcal{X}_{\text{IAMB}}$ with $\epsilon = 0.04$ managed to reach a significant better score than without DPED.

	without DPED	$\epsilon = 0.01$	$\epsilon = 0.04$
$\mathcal{X}_{\text{hiton}}$	0.5150	0.6117	0.7101
$\mathcal{X}_{\text{IAMB}}$	0.5266	0.5003	0.6886
$\mathcal{X}_{\text{hiton filtered}}$	0.5196	0.6275	0.7820
$\mathcal{X}_{\text{IAMB filtered}}$	0.5315	0.6275	0.6275
$\mathcal{X}_{\text{best 21}}$	0.5216	0.5033	0.7140

Table 6.3: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) for different subsets obtained by applying the DPED algorithm with the max increment mode and different values of ϵ to five different subsets for REGED2

The best score for REGED2 was obtained with the subset $\mathcal{X}_{DPED}$ computed with $\mathcal{X}_{\text{best 21}}$, $\epsilon = 0.05$, the max increment mode and the REGED1 dataset as the experimental set instead of REGED2 as it should be. The score reached the value of 0.8284. This setup corresponds to the graph shown in Figure 6.6(a). We can see that there are six good features for the REGED1 dataset. If we analyze a little bit deeper the features in $\mathcal{X}_{DPED}$, we noticed that there were the two parents and four features which are not members of the Markov Boundary. It appeared that these four variables are not manipulated even in the REGED2 dataset, explaining why we could reach such a good score on REGED2 with a subset found with REGED1.

6.2.3 DCED and results

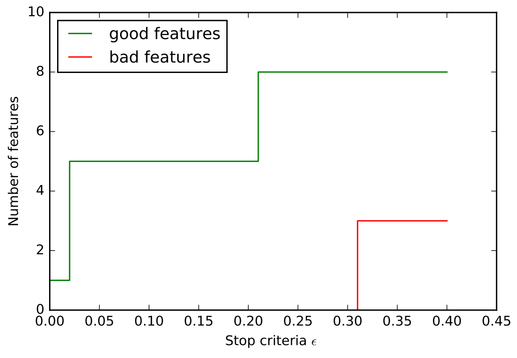
In this section, we will present the results on REGED using the DCED algorithm with the same input subset as presented in the DPED section. We followed the same outline and we also used a Random Forest Classier from Scikit Learn with 1000 estimators with all the other parameters left by default to predict the scores. We only used the Spearman correlation coefficient because it appeared that the Pearson correlation coefficient was slightly less efficient. Moreover, the Spearman coefficient is more versatile since it does not made any assumption about the linearity of the correlation.

The DCED algorithm needs an observational and an experimental set. We used the learning set and REGED1 for this section and REGED2 for the next section. Like the DPED algorithm, the most important parameter is ϵ. The last mode was only pertinent for $\mathcal{X}_{\text{best 21}}$ therefore we focused on the max increment mode for all the other subsets. All the dependencies entailed in this section are between variables of the Markov Boundary through their dependencies on the target. Parents and children are dependent on each other because their are both correlated to the target. Indeed two variable considered independent on the target can be dependent on each other.

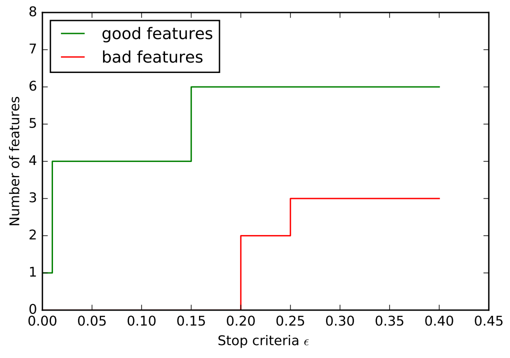
The DCED algorithm is very sensitive to independent features and so to errors in the output subsets of the causal discovery algorithms. One of the best way to remove independent features is to use a filter feature selection but this is not the panacea and even with filtering, this algorithm efficiency relies on the efficiency of the causal discovery algorithms. For example, an independent feature have a very low score and the increment between this variable and the first dependent variable is huge compared to the increment between the scores of unmanipulated and manipulated variables of the Markov Boundary. It has for effect to push out all the variables except the independent ones.

6.2.3.1 REGED1

REGED1 is a lowly manipulated dataset allowing a good efficiency of the DCED algorithm contrary to REGED2. Since both $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{IAMB}}$ are really good approximation of the Markov Boundary without too many errors, we are expecting the filtering to have limited beneficial impact. Indeed, between Figures 6.10(a) and 6.10(b), there are no major improvements with filtering. On the contrary, the filtering reduced the number of good features in the output subset and also decreased the robustness of the method against bad features. The first bad feature for $\mathcal{X}_{\text{hiton}}$ is included around $\epsilon = 0.31$ while for $\mathcal{X}_{\text{hiton filtered}}$ the first bad feature is included around $\epsilon = 0.20$.



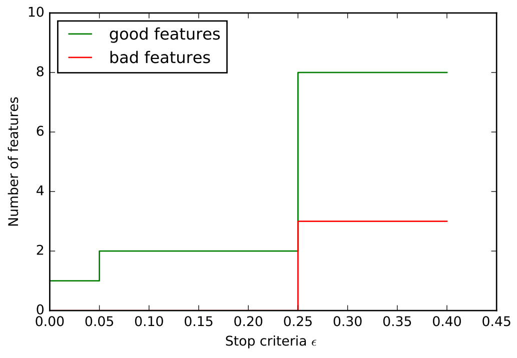
(a) With $\mathcal{X}_{\text{hiton}}$ as input



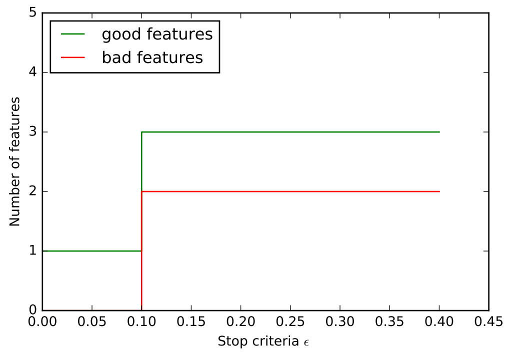
(b) With $\mathcal{X}_{\text{hiton filtered}}$ as input

Figure 6.10: Evolution of the number of good and bad features in $\mathcal{X}_{DCED}$ as a function of ϵ with $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ as input of the DCED algorithm with the max increment mode for REGED1

The conclusions for the subsets from IAMB algorithm are mainly the same than for the subsets from HITON PC algorithm. In Figure 6.11(a) and 6.11(b), we can see that the first features are good features but the next iteration includes some bad features for both $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ with a worse evolution for $\mathcal{X}_{\text{IAMB filtered}}$.



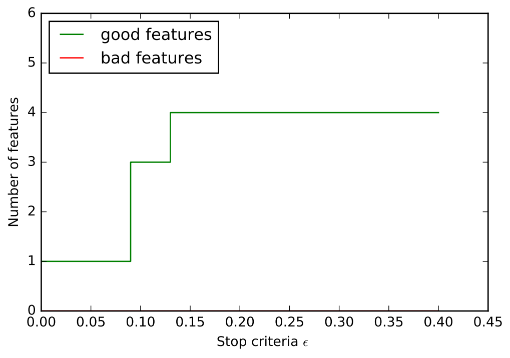
(a) With $\mathcal{X}_{\text{IAMB}}$ as input



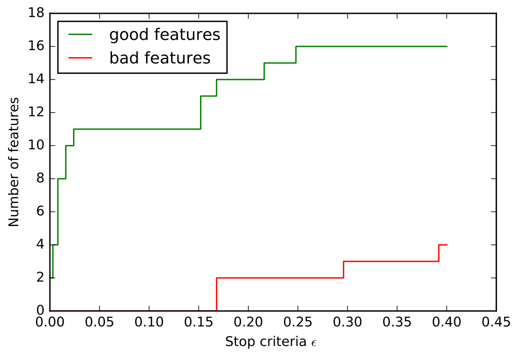
(b) With $\mathcal{X}_{\text{IAMB filtered}}$ as input

Figure 6.11: Evolution of the number of good and bad features in $\mathcal{X}_{DCED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ as input of the DCED algorithm with the max increment mode for REGED1

We were expecting bad performance of the DCED algorithm on $\mathcal{X}_{\text{best 21}}$ because there are a lot of features which are not direct parents or children of the target. The impact of these features is not clear on the graph shown in Figures 6.12(a) and 6.12(b) because these features are not manipulated. Indeed, even if they are less predictive than the set of parents/children, they are considered as good features. The real impact can be seen in Table 6.5 with scores around 0.5 for $\mathcal{X}_{\text{best 21}}$ with the max increment for all the values of ϵ. The last mode was able to counterbalanced this effect since this mode is not looking for a particular set of features with very low score but remove the feature with the highest score until the maximal increment reaches the threshold. The scores with the last mode for $\mathcal{X}_{\text{best 21}}$ are shown in Table 6.4.



(a) With max increment mode



(b) With last mode

Figure 6.12: Evolution of the number of good and bad features in $\mathcal{X}_{DCED}$ as a function of ϵ with $\mathcal{X}_{\text{best 21}}$ as input of the DCED algorithm with the max increment and last mode for REGED1

last mode	without DCED	$\epsilon = 0.1$	$\epsilon = 0.25$
$\mathcal{X}_{\text{best 21}}$	0.5653	0.8835	0.7435

Table 6.4: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) for different subsets obtained by applying the DCED algorithm with the last mode and different values of ϵ to $\mathcal{X}_{\text{best 21}}$

We were able to reach some really good score with DCED especially for very low values of ϵ. The scores obtained with $\mathcal{X}_{\text{hiton}}$, $\mathcal{X}_{\text{hiton filtered}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ are computed with only one feature to learn the model. The scores started to drastically decrease when bad features are included.

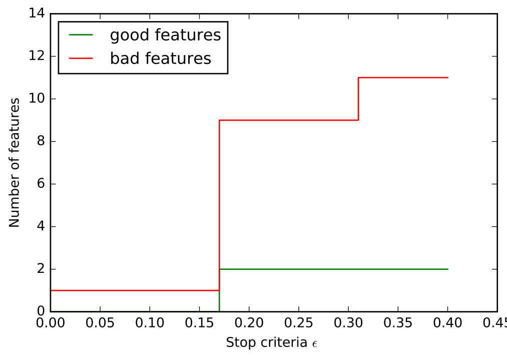
max increment mode	without DCED	$\epsilon = 0.01$	$\epsilon = 0.1$	$\epsilon = 0.25$
$\mathcal{X}_{\text{hiton}}$	0.5789	0.9735	0.9205	0.9343
$\mathcal{X}_{\text{IAMB}}$	0.5749	0.5807	0.8019	0.8019
$\mathcal{X}_{\text{hiton filtered}}$	0.5706	0.9729	0.9216	0.7989
$\mathcal{X}_{\text{IAMB filtered}}$	0.5609	0.9735	0.9735	0.5606
$\mathcal{X}_{\text{best 21}}$	0.5653	0.4998	0.5004	0.4996

Table 6.5: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) for different subsets obtained by applying the DCED algorithm with the max increment mode and different values of ϵ to five different subsets

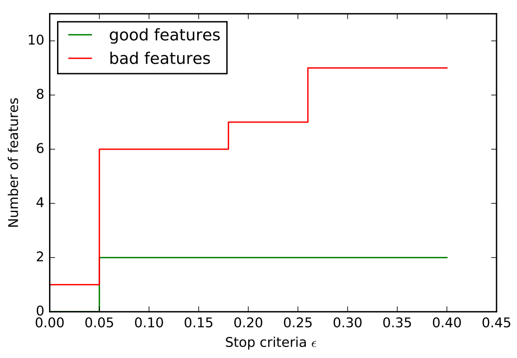
6.2.3.2 REGED2

As explained in the limitations section of the DCED algorithm, it is really hard to find the parents when all the children are manipulated because the correlation between the parents is not systematic. If they are correlated in the observational dataset and one of the parent is manipulated in the experimental dataset, the DCED algorithm tends to reject the manipulated parents because there is a variation of the correlation between the two datasets. Moreover, if there are only two parents, the algorithm is not able to distinguish which one is manipulated and rejects both parents.

In Figure 6.13, we can see the evolution of the number of features in $\mathcal{X}_{DCED}$. The first included feature is a bad feature and when the good features are added to $\mathcal{X}_{DCED}$ the algorithm also includes a lot of bad features. The first included feature is a lowly correlated feature. Hence, its score is very low and the algorithm selects it for a wide range of ϵ before starting to include an other subset of features which have almost the same score. It is important to notice that the two parents have exactly the same behavior than a lot of bad features. It is really hard to find them in this context.



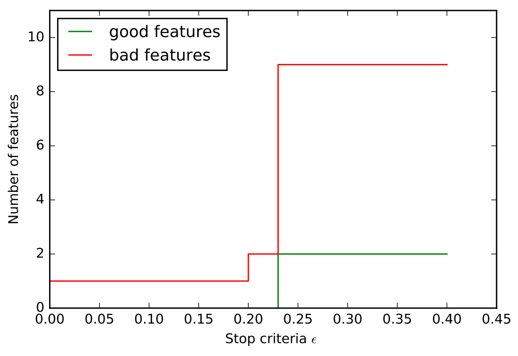
(a) With $\mathcal{X}_{\text{hiton}}$ as input



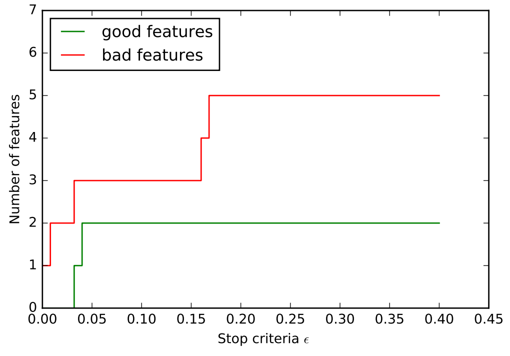
(b) With $\mathcal{X}_{\text{hiton filtered}}$ as input

Figure 6.13: Evolution of the number of good and bad features in $\mathcal{X}_{DCED}$ as a function of ϵ with $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{hiton filtered}}$ as input of the DCED algorithm with the max increment mode for REGED2

The evolution of the number of features in $\mathcal{X}_{DCED}$ is roughly the same for the subsets obtained with HITON PC and the subsets obtained with IAMB. Nevertheless, with $\mathcal{X}_{\text{IAMB filtered}}$, we can notice that the number of bad features are slightly lower than for the three other subsets as it can be seen in Figure 6.14.



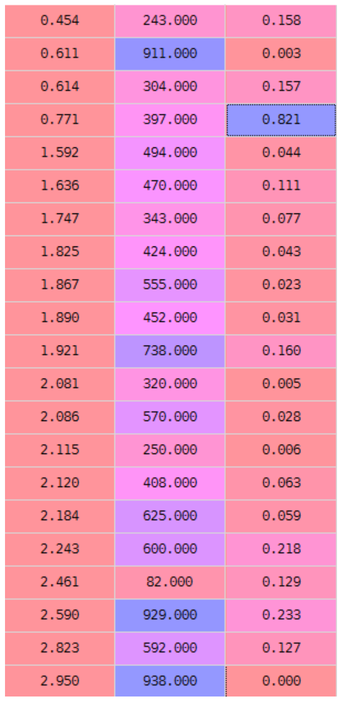
(a) With $\mathcal{X}_{\text{IAMB}}$ as input



(b) With $\mathcal{X}_{\text{IAMB filtered}}$ as input

Figure 6.14: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB}}$ and $\mathcal{X}_{\text{IAMB filtered}}$ as input of the DCED algorithm with the max increment mode for REGED2

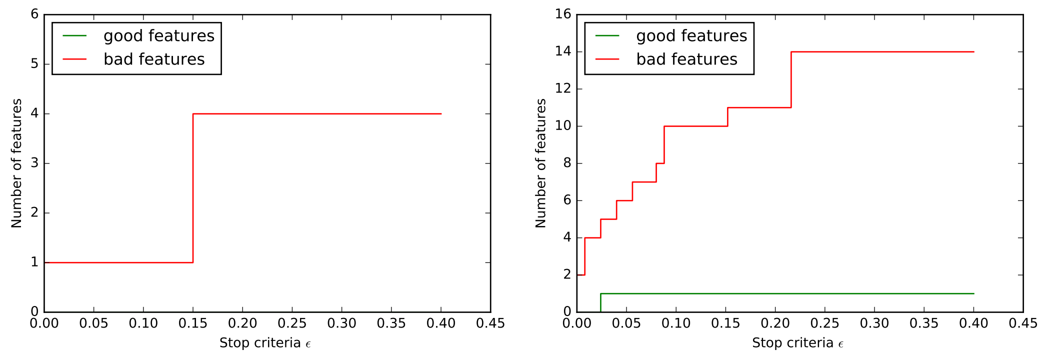
As said previously, the DCED algorithm has poor performances when there are some non-parent and non-children features in the input subset because these features do not behave like the parents and the children and so lead to wrong conclusion. A good example of the limitations described in Section 4.3.3 of the DCED algorithm is shown in Figure 6.16(a). Indeed, some features have such low scores that they literally push the other features out. We can clearly see in Figure 6.15 that the four first features has significantly lower score than the others although they are neither a parent nor a children.



0.454	243.000	0.158
0.611	911.000	0.003
0.614	304.000	0.157
0.771	397.000	0.821
1.592	494.000	0.044
1.636	470.000	0.111
1.747	343.000	0.077
1.825	424.000	0.043
1.867	555.000	0.023
1.890	452.000	0.031
1.921	738.000	0.160
2.081	320.000	0.005
2.086	570.000	0.028
2.115	250.000	0.006
2.120	408.000	0.063
2.184	625.000	0.059
2.243	600.000	0.218
2.461	82.000	0.129
2.590	929.000	0.233
2.823	592.000	0.127
2.950	938.000	0.000

Figure 6.15: Scores of the features in $\mathcal{X}_{\text{best 21}}$ with the DCED algorithm and the max increment mode for the first iteration. The first column is the score, the second is the id of the features and the last column is the increment between two consecutive features

The last mode can decrease the negative impact of such features but have limited effects as it can be seen in Figure 6.16(b).



(a) With max increment mode (b) With last mode

Figure 6.16: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{best 21}}$ as input of the DCED algorithm with the max increment and last mode for REGED1

None of the presented subsets with the DCED algorithm were able to retrieve the good features and always included a relatively high number of bad features leading to bad predictions and low scores. All the scores are reported in the Table 6.6. The only subset with an improvement with DCED is $\mathcal{X}_{\text{IAMB filtered}}$. As said before, it is the only subset limiting the number of bad features in $\mathcal{X}_{DCED}$ for low values of ϵ.

max increment mode	without DCED	$\epsilon = 0.05$	$\epsilon = 0.25$
$\mathcal{X}_{\text{hiton}}$	0.5150	0.5036	0.5320
$\mathcal{X}_{\text{IAMB}}$	0.5266	0.4953	0.5409
$\mathcal{X}_{\text{hiton filtered}}$	0.5196	0.4997	0.5434
$\mathcal{X}_{\text{IAMB filtered}}$	0.5315	0.6056	0.5315
$\mathcal{X}_{\text{best 21}}$	0.5216	0.5033	0.5006

Table 6.6: Scores of the predictions made with a Random Forest algorithm(n_estimators=1000) for different subsets obtained by applying the DCED algorithm with the max increment mode and different values of ϵ to five different subsets for REGED2

6.3 P1000

In this section, we reported the results for the P1000 dataset. We focused our analysis on the two subsets $\mathcal{X}_{\text{IAMB1}}$ and $\mathcal{X}_{\text{IAMB2}}$ described in the last chapter. We first showed the results of a filtering on these subsets before applying the DPED and DCED algorithms.

All the subsets are used to learn some models to predict the target. We used the mean absolute error to assess the performance of each subset. Since the P1000 dataset is a regression problem, we used a Random Forest Regressor from Scikit Learn with 1000 estimators while the other parameters are left by default to predict the target.

A brief description of the two subsets used in this section can be find below :

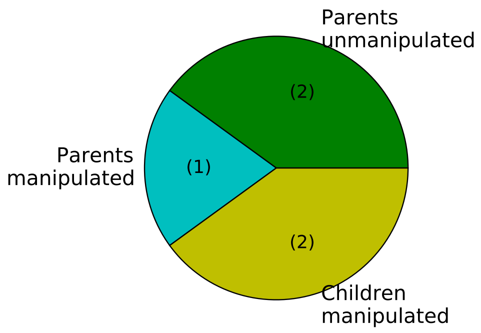
- $\mathcal{X}_{\text{IAMB1}}$ is the output subset of the algorithm IAMB with $\alpha = 0.0002$
- $\mathcal{X}_{\text{IAMB2}}$ is the output subset of the algorithm IAMB with $\alpha = 0.005$

In this dataset, both children are manipulated and the good features only contain the parents and all the unamnipulated features not included in the Markov Boundary since there are no spouses.

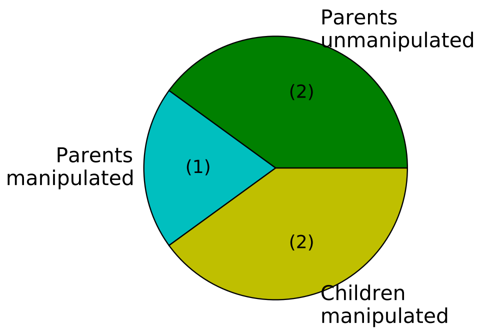
6.3.1 Features filtering in the approximated Markov Blanket

We used the same function as for REGED (*feature_filtering*) but we replaced the Random Forest Classifier by a Random Forest Regressor which also implements a *feature_importances_* attributes. We arbitrarily chose $k = 40$. Since the subset $\mathcal{X}_{\text{IAMB1}}$ is an exact approximation of the true Markov Boundary, the filtering was not really useful but we can highlight the fact that it does not affect the composition of the subset (see

Figure 6.17).



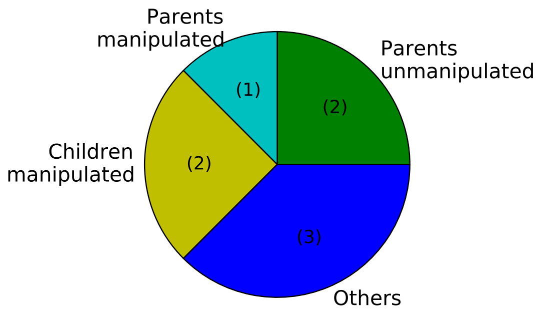
(a) Nature of the features in $\mathcal{X}_{\text{IAMB1}}$ with manipulations labels



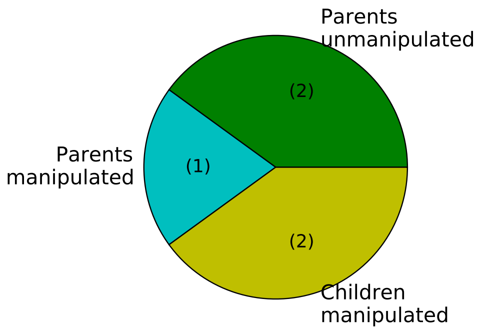
(b) Nature of the features in $\mathcal{X}_{\text{IAMB1}}$ with manipulations labels after filtering

Figure 6.17: Nature of the features of the subset $\mathcal{X}_{IAMB\_1}$ before and after filtering with the **40** best features ranked with the *feature_importances_* attribute of a Random Forest regressor (max_features = None, n_estimators = 1000) learned on all the features from the observational dataset for P1000

On the contrary, there are three false positives features in $\mathcal{X}_{\text{IAMB2}}$. These features are part of the added independent variables (see description of the P1000 dataset in Section 3.3). Therefore they perfectly fit in the targeted features that we try to remove by filtering because they can be very harmful to the efficiency of DPED and DCED as we will show in the next section. In Figure 6.18, we can see that the filter manages to remove the false positive features and the output subset $\mathcal{X}_{\text{IAMB2 filtered}}$ is exactly the same as $\mathcal{X}_{\text{IAMB1}}$ or the true Markov Boundary.



(a) Nature of the features in $\mathcal{X}_{\text{IAMB2}}$ with manipulations labels



(b) Nature of the features in $\mathcal{X}_{\text{IAMB2}}$ with manipulations labels after filtering

Figure 6.18: Nature of the features of the subset $\mathcal{X}_{\text{IAMB2}}$ before and after filtering with the **40** best features ranked with the *feature_importances_* attribute of a Random Forest model (max_features = None, n_estimators = 1000) learned on all the features from the observational dataset for P1000

6.3.2 DPED and results

From the previous section, after filtering, we realized that $\mathcal{X}_{\text{IAMB1}}$, $\mathcal{X}_{\text{IAMB1 filtered}}$ and $\mathcal{X}_{\text{IAMB2 filtered}}$ are exactly the same and correspond to the Markov Boundary. Hence, we only showed the results for $\mathcal{X}_{\text{IAMB1}}$ and $\mathcal{X}_{\text{IAMB2}}$. There is no major differences between the max increment and the last mode except for very low values of ϵ as it can be seen in Figure 6.19. The graph are likely to be similar because of the small size of $\mathcal{X}_{\text{IAMB1}}$. The DPED algorithm managed to discover all the good features with both mode. These good features are the parents of the target and so they are the optimal subset for the manipulated dataset.

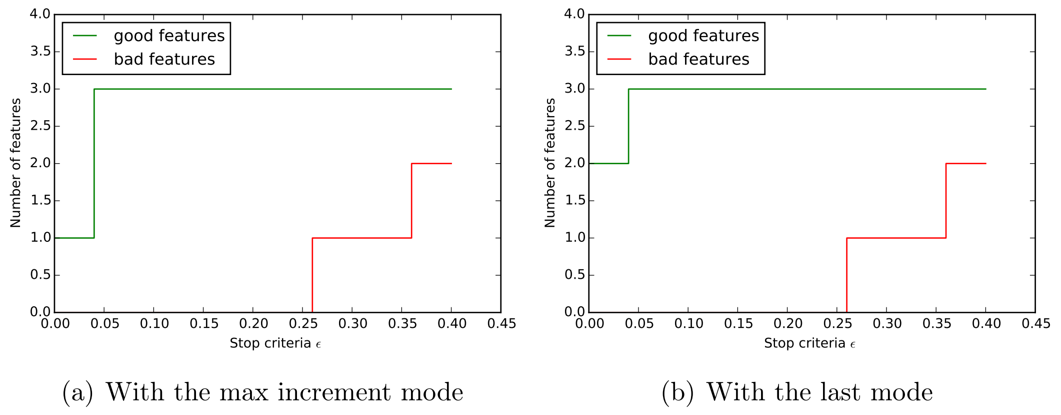
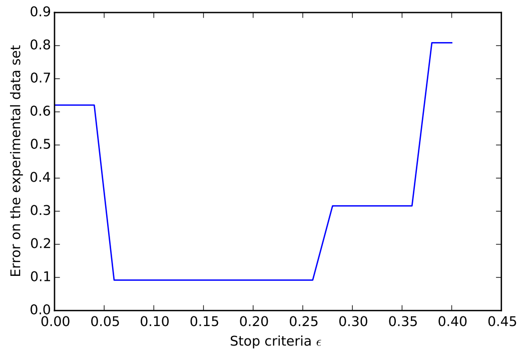
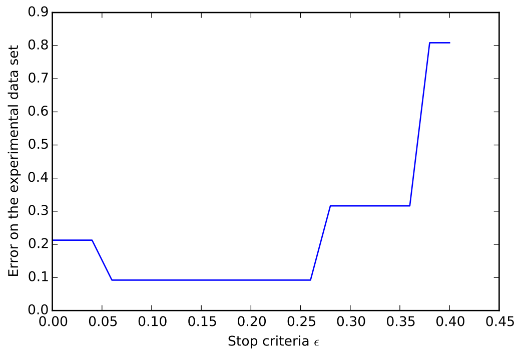


Figure 6.19: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB1}}$ as input of the DPED algorithm with the max increment and the last mode for the experimental set

Intuitively, we can assume that the best performances and so the lowest errors are reached when $\mathcal{X}_{DPED}$ contains only the three parents. We can see in Figure 6.20 that the value of 0.9 which is the lowest error is reached between $\epsilon = 0.05$ and $\epsilon = 0.26$ where $\mathcal{X}_{DPED}$ is only composed of the parents. We can also notice the big negative impact of the bad features on the error.



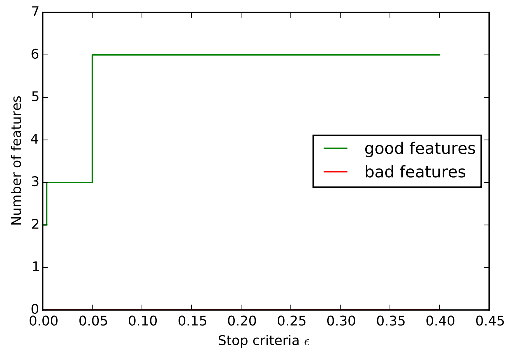
(a) With the max increment mode



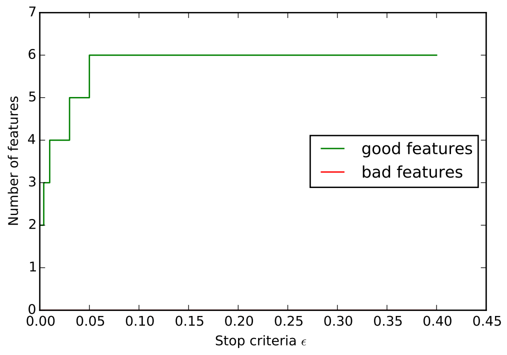
(b) With the last mode

Figure 6.20: Evolution of the error on the prediction for the experimental set as a function of ϵ with $\mathcal{X}_{\text{IAMB1}}$ as input of the DPED algorithm with the max increment and the last mode

Figures 6.21 and 6.22 perfectly illustrate the effect of lowly predictive variable on the evolution of the nature of the features in $\mathcal{X}_{DPED}$ as a function of ϵ. For low values of ϵ, only the independent features are included in $\mathcal{X}_{DPED}$ leading to high errors of prediction even if they are considered as good features since they are not manipulated. They literally push the good and highly predictive features out of the first group of selected features. We need to use an higher value of ϵ to include the parents in this case. They delayed the detection of the parents because of their very low scores. On the other hand, they also pushed the bad features. Hence, the minimal value of ϵ to include the first bad feature is around 0.9. It is hard to be categorical about their impact on DPED because they have both negative and positive effects even if the positive effects are not well defined and could be not recurrent. There are no big differences between the max increment and the last mode.

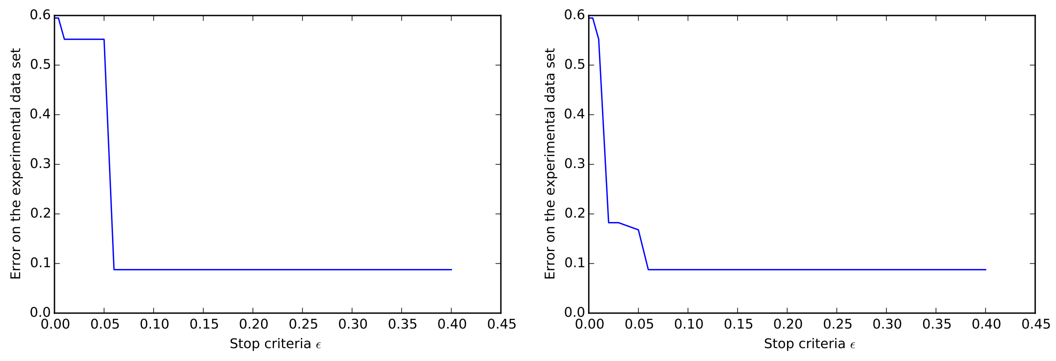


(a) With the max increment mode



(b) With the last mode

Figure 6.21: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB2}}$ as input of the DPED algorithm with the max increment and the last mode for the experimental set

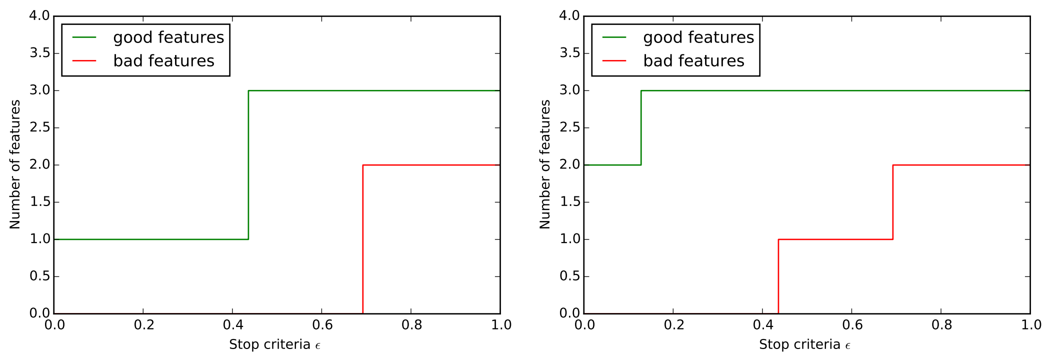


(a) With the max increment mode (b) With the last mode

Figure 6.22: Evolution of the error on the prediction for the experimental set as a function of ϵ with $\mathcal{X}_{\text{IAMB2}}$ as input of the DPED algorithm with the max increment and the last mode

6.3.3 DCED and results

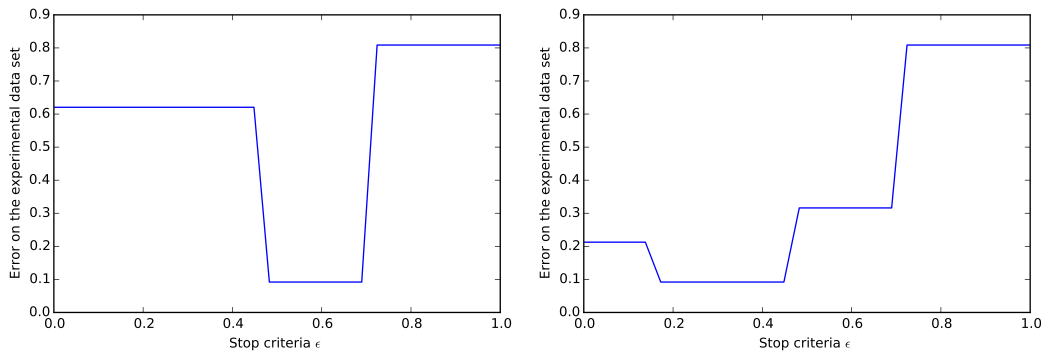
In this section, we showed the results obtained with the DCED algorithm for $\mathcal{X}_{\text{IAMB1}}$ and $\mathcal{X}_{\text{IAMB2}}$. We only used the Spearman correlation coefficient. When the size of the input subset of features is small, the last mode has better performances because it ensures to check the impact of each feature on the score of the others. The graphs in Figure 6.23 show that the last mode managed to find two out of the three parents immediately while with the max increment mode the DCED algorithm only found one parent. Moreover, in Figure 6.23(b), we can see that the third parent is included quite fast. The last mode also started to include bad features for lower values of ϵ than the max increment mode.



(a) With the max increment mode (b) With the last mode

Figure 6.23: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB1}}$ as input of the DCED algorithm with the max increment and the last mode for the experimental set

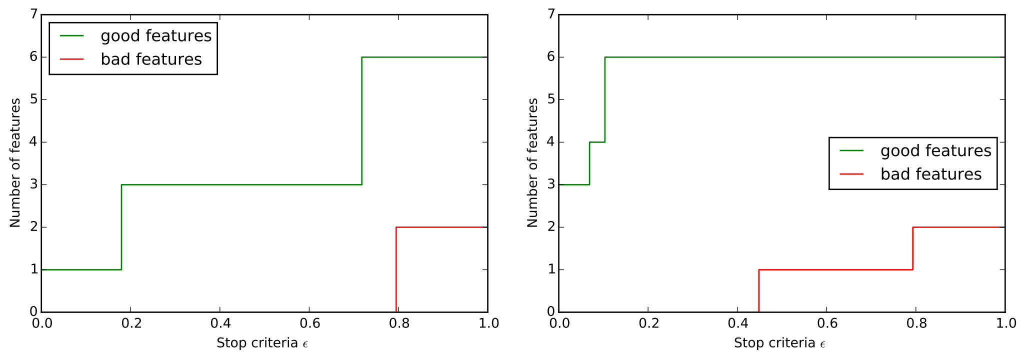
The differences between the error graph for both modes highlight the robustness of the last mode compared to the max increment mode (Figure 6.24). Indeed, they both were able to retrieve the optimal subset but the range of values for ϵ is smaller for the max increment mode. More important, outside of this range, the max increment mode has a really bad performance while the last mode still has good prediction even if $\mathcal{X}_{DPED}$ is not the optimal subset.



(a) With the max increment mode (b) With the last mode

Figure 6.24: Evolution of the error on the prediction for the experimental set as a function of ϵ with $\mathcal{X}_{\text{IAMB1}}$ as input of the DCED algorithm with the max increment and the last mode

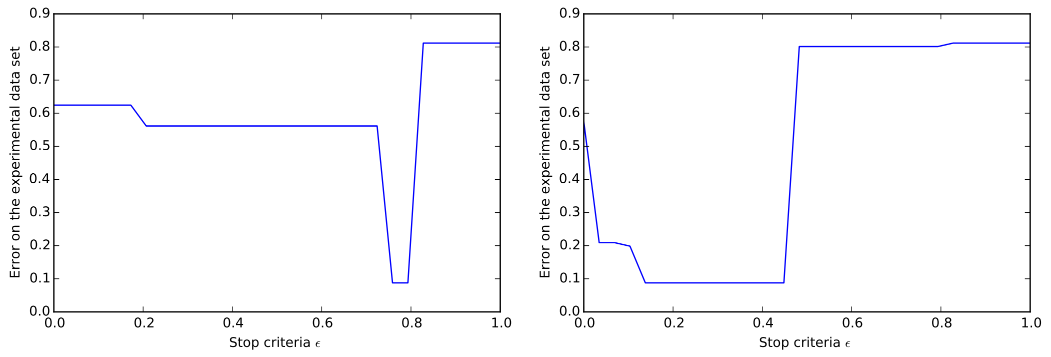
With $\mathcal{X}_{\text{IAMB2}}$, the negative impact of the independent features can clearly be shown especially for the max increment mode. For this mode, the parents could be found but only for a small range of ϵ and not near zero making harder the recognition of this range as it can be seen in Figures 6.25(a) and 6.26(a). The last mode was able to counterbalance the negative impact of the independent features making easier to find the optimal subset (Figures 6.25(b) and 6.26(b)). For $\mathcal{X}_{\text{IAMB2}}$, the optimal subset is composed of the six good features because it was impossible to find the three parents with the DCED algorithms without including the three independent features.



(a) With the max increment mode (b) With the last mode

Figure 6.25: Evolution of the number of good and bad features in $\mathcal{X}_{DPED}$ as a function of ϵ with $\mathcal{X}_{\text{IAMB2}}$ as input of the DCED algorithm with the max increment and the last mode for the experimental set

The evolution of the error computed with $\mathcal{X}_{DCED}$ found with the max increment mode and $\mathcal{X}_{\text{IAMB2}}$ is shown in Figure 6.26(a). We can see that the smallest error only appears for a small range of ϵ while for the rest of the graph the error is very high. With the last mode, the range of values for ϵ with the same lowest error is wider than the range with the max increment mode.



(a) With the max increment mode (b) With the last mode

Figure 6.26: Evolution of the error on the prediction for the experimental set as a function of ϵ with $\mathcal{X}_{\text{IAMB2}}$ as input of the DCED algorithm with the max increment and the last mode

We can assess the robustness of these algorithms by looking on the range of values for ϵ for which they were able to retrieve a good set of features for the prediction. In the case of the P1000 dataset, the DCED algorithm was less robust than the DPED algorithm but still managed to find a very good subset of features even if it is harder to find an optimal value of the parameter ϵ.

6.4 Chandran and Singh

In this section, we described the results obtained with the Chandran and Singh datasets. We did not have any information about the local graph or their distribution. We applied the three stages previously described to find a subset of features with only predictive features. We used Singh as the observational dataset and Chandran as the experimental dataset. Therefore we considered that we had access to the target of Singh but not to the target of Chandran. The scores obtained on Chandran through all stages are shown in Table 6.7. The scores are all computed with a Random Forest Classifier with 1000 estimators and the other parameters are left by default.

We started by computing a reference score with all the features. Then we tried to apply a classical feature selection based on the Random Forest *features_ importances_* attribute ranking. We selected the 20 best features to build the subset $\mathcal{X}_{\text{best } 20}$. Then, we decided to use a local causal algorithm to find a new subset of features. We chose the HITON PC algorithm with $\alpha = 0.05$ and we called the output subset $\mathcal{X}_{\text{hiton}}$. With the two first feature selection, the scores slightly decreased.

The second stage consisted in applying a filter to $\mathcal{X}_{\text{hiton}}$. We compared the 40 best features from Random Forest ranking with the features in $\mathcal{X}_{\text{hiton}}$ and only selected the features in both subsets. We called this new subset $\mathcal{X}_{\text{hiton filtered}}$. We finally managed to improve the performance of the prediction.

Features selected	score
All features	0.7448
$\mathcal{X}_{\text{best 20}}$	0.7144
$\mathcal{X}_{\text{hiton}}$	0.7060
$\mathcal{X}_{\text{hiton filtered}}$	0.7990

Table 6.7: Scores of the predictions made with a Random Forest Classifier(n_estimators=1000) for different subsets of features for Chandran

The third stage was to apply the DPED algorithm to $\mathcal{X}_{\text{hiton filtered}}$ to try to remove all the bad features that we could not find before. The evolution of the score as a function of the threshold ϵ is shown in Figure 6.27. We can notice that the best score is around 0.85 and so an increase of 0.1 between the score with all the features and the best score.

The size of the output subset of the DPED algorithm $\mathcal{X}_{DPED}$ for the optimal value of ϵ is composed of only two features. It means that we managed to increase the score by 0.1 while reducing by a factor 6000 the size of the input subset. This is a very important point when we want to limit the number of experimentation without loss of accuracy.

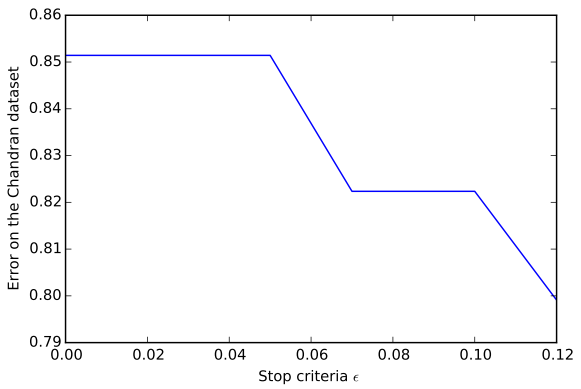


Figure 6.27: Evolution of the score on the prediction for the Chandran dataset as a function of ϵ with $\mathcal{X}_{\text{hiton filtered}}$ as input of the DPED algorithm with the last mode

6.5 conclusion

The conclusion for both REGED and P1000 dataset are likely the same even if they present slightly different situations. On REGED, some manipulated features were hard to find due to their low performance when used alone to learn a model while on P1000 the major problem was the independent features in the input subset.

On overall, both DPED and DCED had significantly positive impact on the prediction of the target for all the manipulated dataset (REGED1,REGED2 and P1000 experimental dataset). We also noticed that both algorithms showed their limits on heavily manipulated dataset like REGED2. Indeed, they were not able to isolate the two parents with a proper setting even if we found a subset with the two parents and four unmanipulated external variables but with the wrong setting. This performance was realized with the wrong dataset (we used the dataset REGED1 as input of the DPED algorithm but we used the output subset to predict on REGED2).

The filtering applied to $\mathcal{X}_S$ (the input subset of tested features) was really important for P1000 to remove the independent features but had mitigated results for REGED because the filtering mainly removed good features since both $\mathcal{X}_{\text{hiton}}$ and $\mathcal{X}_{\text{IAMB}}$ were really good approximation of the Markov Boundary. It appeared that the removed features were predictive enough and so should be kept. Therefore, it was important to set the parameter k with a value large enough to avoid to remove them. In that case, the problem was to find a correct setting describing the highly and lowly predictive features. When the features are independent, it is easy to retrieve them because they are not predictive at all but if we want to treat lowly predictive features we need to define what is lowly. This is done through the parameter k.

We only used Random Forest algorithms in this chapter but they were really time consuming especially for the DPED since the number of learned models is proportional to $(k + 1) * n$ with k the number of fold and n the number of features in $\mathcal{X}_S$. From the computation time point of view, the DCED algorithm was much faster than DPED because the computation time of the Spearman correlation was lower than the fitting time of a Random Forest predictor.

Finally, from all the results of the chapter 6, we can conclude that including a bad feature is more harmful than omitting a good feature in the subset used to predict the target. Therefore, it is sometimes more interesting to use a lower value of ϵ to ensure to have only good features in the final susbet.

Chapter 7

Conclusion and perspectives

In this master's thesis, we proposed a three stage process to select an optimal subset of features for predicting a manipulated dataset :

1. Select relevant feature subset $\mathcal{X}_S$ from observational data (eg. Using svm, random forest or causal methods)

2. Filter out lowly predictive features from $\mathcal{X}_S$ such as independent features

3. Filter out manipulated non-parent features from $\mathcal{X}_{S\text{filtered}}$ using both observational and experimental data.

The stage 3 involved two newly developed algorithms (DPED and DCED) to extract information from both observational and experimental dataset. These algorithms were shown to be quite effective if the number of manipulations is not too large (see REGED2). The last stage was focus on predicting the target of the experimental test set with a model learned on the subset found in stage 3. In the framework of this master thesis, we limited our analysis to the Random Forest predictor thanks to its efficiency.

The algorithms developed during this master's thesis are detailed in Chapter 4 where the motivation and the implementation are described. Their theoretical limitations are also exposed. It appeared that the most sensitive parameter is the input subset. Therefore, the efficiency of the stage 1 and 2 is very important to reach good performance.

Chapter 5 detailed the results obtained in stage 1 as well as their prediction performance to highlight the limitations of classical feature selection. We also compared the classical and causal approach about the Markov Boundary discovery. Finally, we selected some representative subset of features to process in stage 2.

Chapter 6 presented the results obtained when we tried to extract information from the manipulation to improve the performances on the experimental test set. This chapter corresponds to the stage 2, 3 and the last stage. We first showed the impact of filtering on the subsets before applying the filtered subset $\mathcal{X}_{S\text{filtered}}$ to DPED and DCED. We highlighted the difficulties to find a good definition of a lowly predictive feature with the aid of the parameter k (the number of considered features in the ranking). We assessed the efficiency of DPED and DCED to retrieve a good subset of features as a function of the main parameter ϵ. Finally, we used some output subsets found at the stage 3 to predict the targets of the respective experimental test set. A clear improve of the prediction was noticed especially for REGED1 and P1000. The heavily manipulated dataset REGED2 was really hard to process but we managed to obtain quite good results for a particular setting.

On the real data represented by the datasets Chandran and Singh, the DPED algorithm managed to improve the predictive efficiency while drastically reducing the number of selected features. This results is really rewarding because the DPED algorithm was mainly tested during its development on artificial datasets.

7.1 Perspectives and further improvements

7.1.1 DPED

We restricted our analysis to the Random Forest algorithm but it could be interesting to use other learning algorithms to train all the inner models of the DPED algorithms.

The score computed to rank the features is the mean absolute error because it can be used for both classification and regression but we think that other metrics such as balanced accuracy ($0.5 * (specificity + sensitivity)$) for classification can be very efficient.

The parameter ϵ has a huge impact on the final result and it could be very useful to implement an automatic way to stop the iterative process when the scores of the features are under a certain threshold . The basic idea would be to shuffle the features in the manipulated dataset to simulate a full manipulated distribution and compute a statistically representative value of the score over which the features are considered as bad features.

7.1.2 DCED

In this master thesis, we focused on the Spearman and Pearson correlations but a deeper comparison with other correlation statistical tests should be done like χ^2 test.

The DCED algorithm is based on the same criteria as DPED to stop the iterative process. Currently, we let the user set this parameter and it could be interesting to implement an automatic process to find the best value of the threshold criteria ϵ.

Bibliography

[1] Constantin Aliferis, C. F. Aliferis, Ioannis Tsamardinos, and Alexander Statnikov. Causal explorer:causal probabilistic network learning toolkit for biomedical discovery, 2003.

[2] Constantin Aliferis, Ioannis Tsamardinos, Alexander Statnikov, C. F. Aliferis M. D, Ph. D, I. Tsamardinos Ph. D, and Er Statnikov M. S. Hiton, a novel markov blanket algorithm for optimal variable selection, 2003.

[3] Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation.

[4] Gianluca Bontempi. Machine learning methods for bioinformatics.

[5] Leo Breiman. Random forests.

[6] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification.

[7] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees.

[8] Isabelle Guyon, Constantin Aliferis, and André Elisseeff. Causal feature selection, 2007.

[9] Isabelle Guyon, Constantin F. Aliferis, Gregory F. Cooper, André Elisseeff, Jean-Philippe Pellet, Peter Spirtes, and Alexander R. Statnikov. Datasets of the causation and prediction challenge.

[10] Isabelle Guyon, Constantin F. Aliferis, Gregory F. Cooper, André Elisseeff, Jean-Philippe Pellet, Peter Spirtes, and Alexander R. Statnikov. Design and analysis of the causation and prediction challenge.

[11] DM Hausman and J Woodward. Independence, invariance and the causal markov condition.

[12] Thibault Helleputte and Pierre Dupont. Feature selection by transfer learning with linear regularized models.

[13] C. Aliferis I. Tsamardinos. Towards Principled Feature Selection: Relevance,Filters and Wrappers. 2003.

[14] Ron Kohavi and George H. John. Wrappers for feature subset selection.

[15] Alexander Statnikov, Sisi Ma, Mikael Henaff, Nikita Lytkin, Efstratios Efstathiadis, Eric R. Peskin, and Constantin F. Aliferis. Ultra-scalable and efficient methods for hybrid observational and experimental local causal pathway discovery.

[16] Alexander R. Statnikov, Jan Lemeire, and Constantin F. Aliferis. Algorithms for discovery of multiple markov boundaries.

[17] Ioannis Tsamardinos, Constantin Aliferis, Alexander Statnikov, and Er Statnikov. Algorithms for large scale markov blanket discovery. In In The 16th International FLAIRS Conference, St, 2003.

[18] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy.