UNIVERSITY OF PORTO FACULTY OF ENGINEERING

Deteção de Fraude em Telecomunicações com Técnicas de Data Mining

João Vitor Cepêda de Sousa

Master in Electrical and Computers Engineering

Supervisor: Prof. Dr. Carlos Manuel Milheiro de Oliveira Pinto Soares

Co-Supervisor: Prof. Dr. Luís Fernando Rainho Alves Torgo

July 31, 2014

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES 2013/2014

A Dissertação intitulada

"Deteção de Fraude em Telecomunicações com Técnicas de Data Mining"

foi aprovada em provas realizadas em 25-07-2014

o júri

Presidente Professora Doutora Maria Teresa Magalhães da Silva Pinto de Andrade
Professora Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor Paulo Jorge de Sousa Azevedo
Professor Auxiliar do Departamento de Informática do Universidade do Minho

Professor Doutor Carlos Manuel Milheiro de Oliveira Pinto Soares
Professor Associado do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.

Autor - João Vitor Cepeda de Sousa

Faculdade de Engenharia da Universidade do Porto

Abstract

This document presents the final report of the thesis "Telecommunication Fraud Detection Using Data Mining Techniques", were a study is made over the effect of the unbalanced class data, generated by the Telecommunications Industry, in the performance of the classification algorithm Naive Bayes. In this subject, an unbalanced class data set is characterized by an uneven class distribution where the amount of fraudulent instances (positive class) is substantially smaller than the amount normal instances (negative class). This will result in a classifier which is most likely to classify data belonging to the normal class then to the fraud class. At first, an overall inspection is made over the data characteristics and the Naive Bayes model. A feature engineering stage is executed, with the intent to extend the information contained in the data creating a depper relation with the data itself and algorithm characteristics. A previously proposed solution that consists on undersampling the most abundant class (normal) before building the model, is presented and tested. In the end, the new proposals are presented. The first proposal is to study the effects of changing the intrinsic class distribution parameter in the Naive Bayes model and evaluate its performance. The second proposal consists in estimating margin values that when applied to the model output, attempt to bring more positive instances from previous negative classification. All of these suggested models are validated over a *monte-carlo* experiment, using data with and without the engineered features.

Acknowledgements

My first words of acknowledgment must be to my mother, Olimpia Cepêda, for the love and patient she offered me during my path to become an engineer. She always supported all of my choices and gave me strength to endure it all. I thank her for all the work I gave her during my growing up and I hope to later on help her as much as she did me.

A big and lovely thanks to my woman, Tânia Trindade. Without her love it would be harder to support the long hours of work without breaking. She had the patient and the happiness to cheer me when I was down and to be behind my back when I need it. I hope to be there for all her hard times.

I would also like to thank my supervisor, Carlos Soares, and co-supervisor, Luís Torgo, for all the help and advisement they gave me during this work. In a more funny way to describe it, I thank for all the patient that they have showed when I asked my thousands of silly questions. I will attempt to be there for your assistance when you need or request it.

To finish, I also like to thank my friends for the support, and my laptop for not giving up on my thousand hour's data processing marathons!

Thank you!

João Cepêda

"Prediction is very difficult, especially if it's about the future."

Niels Bohr

Contents

List of Figures

List of Tables

Abbreviations

ASPeCT	Advanced Security for Personal Communications Technologies
CDR	Call Details Record
CFCA	Communications Fraud Control Association
DC-1	Detector Constructor
LDA	Latent Dirichlet Allocation
MF	Multiple-Frequency
SIM	Subscriber Identity Module
SMS	Short Message System
SOM	Self-Organizing Maps
SVM	Support Vector Machine

Chapter 1

Introduction

1.1 Motivation

Telecommunications fraud is one of the major concerns in the telecommunications industry and corresponds to the abusive usage of an operator infrastructure. This means that a third party is using the resources of a carrier without the intention of paying them. This problem results in significal damages in the financial field [1] since fraudsters are currently stealling part of the revenue of the operators who offer these type of services [2, 3]. Others aspects of the problem that causes loss of revenue, are the use of the identity of legitimate clients in order to commit fraud, which result in those clients being framed for a fraud attack that they didn't commit. This will result in the loss of confidence of the client in their carrier, giving them reasons to leave. Besides being victims of fraud, some clients just don't like to work with a carrier who has been victim of fraud attacks. Since the offering of the same services is available by multiple carriers, the client can always switch very easily between them [4].

	2005	2008	2011	2013
Estimated Global Revenue (USD)	1.2 Trillion	1.7 Trillion	2.1 Trillion	2.2 Trillion
Lost Revenue to Fraud (USD)	61.3 Billion	60.1 Billion	40.1 Billion	46.3 Billion
% Representativa	5.11%	3.54%	1.88%	

Table 1.1: Annual Global Telecommunications Industry Lost Revenue

Table 1.1 displays the dimension of the "financial hole" generated by fraud in the telecommunications industry and the impact of fraud in the annual revenue of the operators.

There are a couple of reasons which make the fraud in this area appealing for some fraudsters. One is the difficulty in the tracking of the fraudster. This is very expensive process and requires a lot of time, which makes it impractical to detect a large amount of individuals. Another reason is the technological requirements to commit fraud in these systems. A fraudster doesn't require a particularly sophisticated equipment in order to practice fraud in these systems [5].

All these evidences generate the need to detect the fraudsters in the most effective way in order to avoid future damage to the telecommunications industry.

Data mining techniques are one possible solution, since they allow the identification of fraudulent activity with a certain degree of confidence. Data mining also works specially well in large amounts of data, a characteristic of the data generated by the telecommunications companies.

1.2 Problem

Telecommunication operators store large amounts of data related with the activity of their clients. In these records exists both normal and fraudulent activity records. It is expected for the fraudulent activity records to be substantially less fequent than the normal activity.

The problem that arises when trying to effectively detect fraudulent behaviour in this field is due to the effect that the class distribution (normal or fraud) has over the algorithms which construct the classifiers. The classifier being a model that allows the classification of data as normal or fraudulent, and therefore the main objective of the fraud detection studies.

One problematic example is the impact of the class distribution on the data used for the classifier construction algorithm, Naive Bayes. If the class distribution is highly unbalanced then the Naive Bayes algorithm will estimate a classifier which is most likely to classify each instance belonging to the class which is more abundant. In this scenario, the most abundant class would be the normal class.

1.3 Objectives

The objective of this dissertation is to study the performance of the Naive Bayes classifier in a problem of very unbalanced data sets, concerning anomaly detection in the telecom industry. Another goal is to investigate possible adaptations to the model in order to address the problem of unbalanced classes.

Summarizing, the main objectives of this project are:

1. Evaluate the performance of the Naive Bayes classifier on on unbalanced data set from the telecom industry;
2. Apply existing adaptation to the model and evaluate the performance improvements;
3. Propose a new solution to this problem;

1.4 Document Structure

The present document is divided into 5 chapters: Introduction (1), State of the Art (2), Naive Bayes (3), Study Case (4) and Conclusion (5). In the State of the Art, a review is made on previously studies in the area of Telecommunications fraud, Anomaly detection and Data Mining. In the Naive Bayes chapter, is presented the standard Naive Bayes algorithm, as well as its extensions, the adaptations and the proposed modifications. In the Case Study chapter, is presented the data,

the methodology and the results. In the final chapter, the conclusion, an overview of the project is given and some possibilities for future work.

Chapter 2

State Of the Art

2.1 Telecommunications Fraud

Fraud is commonly described as the intent of misleading others in order to obtain personal benefits [3]. In the telecommunications area, fraud is characterized by the abusive usage of and carrier services without the intention of paying. In this scenario there might emerge two victims:

- **Carrier** — Since there are resources of the carrier that are being used by a fraudster, it exists a direct loss of operability since those resources could be used on a legitimate client. This will also result in the loss of revenue since this usage will not be charged.
- **Client** — When victim of a fraud attack, the client integrity is going to be contested in order to detect if this is really a fraud scenario. If the client is truly the victim, then the false alarm will compromise the confidence that the client erstwhile had in the carrier. Still some clients might even be charged for the services that the fraudster used, therefore destroying even further the confidence of the client on the carrier.

The practice of fraud is usually related to money, but there are some other reasons like political motivations, personal achievements and self-preservation that motivate the fraudsters to commit attacks [3] .

2.1.1 Historical Remarks

Since the beginning of commercial telecommunications, the fraudsters have been causing financial damage to the companies who offered these services [6]. At the start, the carriers did not have the dimension, or the users, that they have nowadays and so the amount of fraud cases wasn't as big. This could mean that the financial damage caused, wasn't as high as it is today, but this isn't actually true. Indeed the amount of frauds was smaller but, the cost associated with a fraud attack in the easlier infrastructure was greater than nowadays. Later on and due to the technological advances, the cost of a fraud attack to the carrier has been decreasing, but conversely, the amount of occurrences have been increasing creating a constant financial damage [3].

	700	900	1100	1300	1500	1700
700		1	2	4	7	
900			3	5	8	
1100				6	9	#
1300					0	
1500						*

Table 2.1: Frequency-Digit correspondence on the Operator MF Pulsing

An interesting story about one of the first types of telecommunication fraud happened in 1957, in the telephony network of AT&T. At this time AT&T was installing their first automatic PBXs from *The Bell Systems* company. To access the control options of the network (e.g. call routing), the operators would send Multiple-Frequency (MF) sound signals in 2400Hz band directly to the telephony line, were all the other calls signalling was passing. Since the control signals could be sent from any point of the network, this allowed users to ear and therefore try to copy those signals. The fraud on the telephone lines that consisted on the replication by the user of those control signals in order to use the network resources, like free calling, was denominated by *phreaking* (*phone* + *freak*). The pioneer in this type of fraud was Joe Engressia, also known as *Joybubbles*. He figured that, by whistling to the microphone in headset of his phone at the right frequency he could establish a call between two users without being charged. Later on another phreaker, John Darper, realized that the whistle offered, at the time, in the cereal box *Captain Crunch*, was able to produce a perfect sound signal at the frequency required to access the control commands of the network. This discovery gave him the nickname of *Cpt. Crunch*.

In the 60s *The Bell Systems* released the paper were the control signal were described [7]. This paper and the previous discoveries made from John Darper and Joe Engressia, allowed for Steve Wozniak and Steve Jobs (yes, the creators of apple) to create a device that allowed the job of *phreaking* to become much easier. The device was named *Blue Box* (2.1), and it was a simple box with a keypad and a speaker that would connect to the microphone on the telephone headset. By pressing a key, the device emitted the corresponding sound signal trough the speaker and therefore into the network allowing to access the control with the previous correct frequency whistling effort. The key to MF conversion is visible in table 2.1. All these discoveries and the invention of the blue box had a purely academic purpose, but this didn't discouraged other people to copy the device and abuse its purpose.

More recently, the telecommunications fraud concerns extended to another all new dimension since the commercialization of the internet and to a whole new pack of services. New types of fraud appeared and the operators have less control over the users. This resulted in an increased difficulty in preventing the problem. Carriers keep investing in new security systems in order to prevent and fight the fraudsters, but the fraudsters don't give up and keep trying to find new ways to bypass those securities. This might results in new types of fraud besides the ones that the security systems could prevent. It is safe to say that there is a everlasting fight between the carriers and the fraudsters. The operators trying to avoid their systems from being attacked and the fraudsters

Figure 2.1: Blue Box Device

trying to detect and abuse the weaknesses on the systems [3].

2.1.2 Fraud Methods

Telecommunications is wide area because it is composed by a variety of services like telephones and internet. Fraud in this area is consequently an extensive subject. There are types of frauds that are specific to one service, like the SIM card cloning fraud which is a fraud that can only be committed in the mobile phone service. On the other hand, there are frauds that cover multiple services like subscription fraud, which is associated to the subscription of services, non regarding the type of service. Next, the fraud methods will be described.

- **Superimposed Fraud** — This kind of fraud happens when the fraudster gains unauthorized access to a client account. In telecommunications, this includes the SIM card cloning fraud that consists in the copy of the information inside the SIM card of the legitimate client to another card in order to use it and access the carrier services. This mean that the fraudster superimposes the client actions pretending to be that person. Despite not being so common lately, it is the fraud who has been most reported in the since its detection. Thus, it has been the object of more studies [8].

- **Subscription Fraud** — This fraud occurs when a client subscribes to a service (e.g. internet) and uses it without the intention of paying. This is the fraud method that was reported as the most frequent in the year of 2013 (see fig. 2.2) [3].

- **Technical Fraud** — This fraud occurs when a fraudster exploits the technical weaknesses of a telecommunications system for his benefit. These weaknesses are usually discovered by the fraudster in a trial and error methodology or by the direct guidance of an employee who is committing internal fraud. Technical fraud in mostly common on newly deployed services, where the chance of existing an error of development is bigger. Some of these

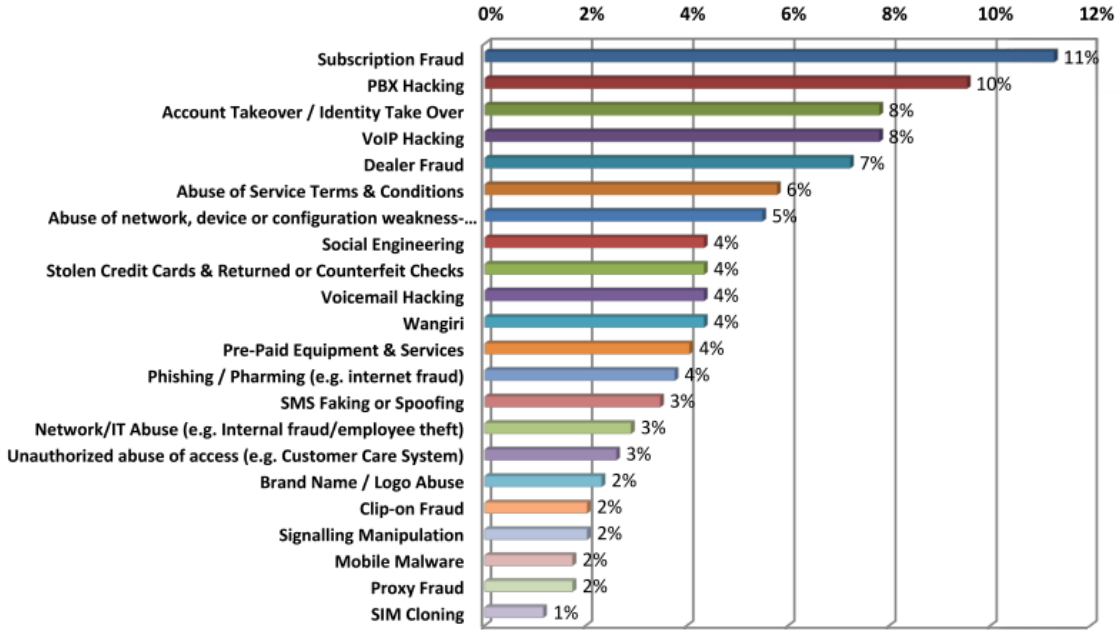

Figure 2.2: 2013 WorldWide Telecommunications Fraud Methods Statistics.

errors are only fixed after the deployment of the system and in some cases after the fraud being committed. This fraud includes the well known "system hacking" [3].

- **Internal Fraud** — Happens when an employee of a telecommunications company uses internal information about the system in order to exploit it for personal benefit or the benefit of others. This fraud is usually related with other types of fraud (e.g. technical fraud).

- **Social Engineering** — Instead of trying to discover the weaknesses of a carrier system (like the trial and error system of technical fraud), the fraudster uses soft skills in order to obtain detailed information about the system. This information can be obtained from the employees of the carrier without them knowing the true intentions of the fraudster. There might even exist a relation between this method and the internal fraud where the fraudster convinces one employee in committing the fraud in his name [3].

Figure 2.2 displat the fraud methods with more incidences in the year 2013 worldwide.

2.2 Anomaly Detection

Anomaly detection, is one of the areas of the application of data mining which consists in the identification of intances of data which do not fit the pattern defined by the data. An anomaly (or outlier) is the instance of data that do not belong to any pattern predefined as normal or in fact belong to a pattern consider abnormal [9]. In summary, the instances belonging to normal patterns are considered normal and those who don't belong are considered abnormal. If those patterns are

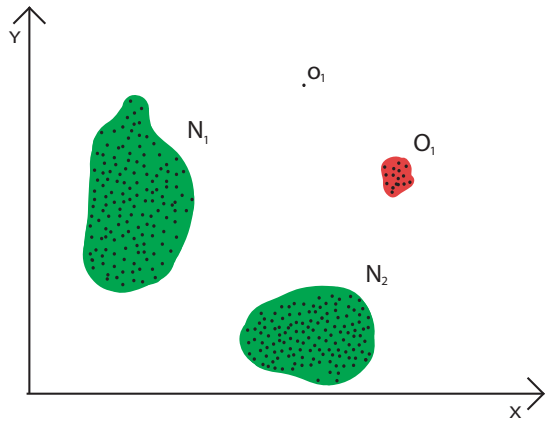

Figure 2.3: Outliers example in a bidimensional data set.

well defined at an early stage and the model used to do the anomaly detection is well built, this process can also be used in the prediction (prevention) of anomalies. Some example of anomaly detection problems are:

- Fraud detection - telecommunications fraud, credit card fraud or insurances fraud.
- Intrusion Detection — network intrusion detection, live house security systems.
- Network Diagnoses — network monitoring, network debugging.
- Image Analyses — image processing.
- Failure Detection — networks failure prediction.

The process used to build an anomaly detection model is composed by two stages, the training stage and the test stage. The first stage is the training stage and is used to build the model used for detection. The second stage is the test stage and is used to evaluate the performance of the model [10]. This process usually implies that the available data set is divided in to parts, one for training and the other for testing. These parts usually represent 70% and 30% of the original data set for training and testing respectively.

Anomaly detection wouldn't be complicated by visual analysis for a small set of data. The problem appears when the amount of data grows exponentially, and the visual analysis starts to be less precise. In figure 2.3 we can observe two patterns of data in the group N_1 e N_2, one smaller group O_1 and an instance o_1. In the case of anomaly detection, the instance o_1 is undoubtedly an outlier and the groups N_1 e N_2 are not outliers. However it can be argued that the smaller group O_1 can be either a group of outliers or just another group of normal data.

2.2.1 Anomaly Types

When building an anomaly detection system, it is necessary to specify the type of anomaly that the system is able to detect [9]. These anomalies can be of type:

- **Point Anomalies** — An individual instance that is considered abnormal relative to the rest of the data. This is the most basic type of anomaly. In figure 2.3 the sample o_1 is considered a point anomaly. An example of a point anomaly in the telecommunications context is when a client that has a mean call cost of value X and suddenly a call of value Y is made $(Y >> X)$. Then due to the difference from the mean, the call of cost Y is very likely to be a point anomaly.

- **Contextual Anomalies** — A data instance that differs from the rest of the data given a certain context (or scenario). The same instance can be classified as normal given a different context. The context definition is defined when formalizing the data mining problem. An example of contextual anomaly in the telecommunications context is when we compare the number of calls made by a user within the business context against personal context. It is more likely for a client to make less calls in the personal context than in the working context. So, if an extraordinary large number of calls appears for that client, it will be considered abnormal in the personal context and normal in the business context.

- **Collective Anomalies** — A group of related data that is considered anomalous according to the rest of the data set. Individual instances in that group may not be considered anomalies when compared to the rest of the data, but when joined together they are considered an anomaly. An example of collective anomaly in the telecommunications context is when the monitoring system that counts the number of active calls in the infrastructure per second is displaying 0 active calls for a 1 minute period measure. This means that 60 measures were made and all of them were returning 0. A minute without any active calls is very likely to be considered an anomaly, but if we take each one of those 60 measures as a single case, or even smaller groups, then they might be considered normal.

2.2.2 Data Mining

Data mining is a process that allows the extraction of non-explicit knowledge from a data set [11]. It consists in the semi-automatic analysis of the data, attempting to find patterns that summarize the relevant information contained in it. This process is very useful for handling big amounts of data, which is a characteristic of the data generated from the telecommunications companies. A typical data mining problem starts from the need to find extra information from an existing set of data. Each step in the process is measured in order to achieve the maximum amount of information for the planned objective. The steps that compose the standard process are:

- **Data cleaning and integration** — Cleaning consist in the removal of noise and data that are consider redundant and useless. In the integration phase, data from multiple sources is combined.

- **Data Selection and Transformation** — In the selection step, only the data that is relevant for the task is selected. This data is prepared for the algorithm used in the data mining process (next step). Transformation allows the creation of new data derived from the previous data, such as new variables that are a combination of the original ones.

- **Data Mining** — This is the step where the new knowledge will be extracted from the data. An algorithm is applied to the data in order to obtain a model that represents the data characteristics. Some algorithms used in this step are described in the anomaly detection chapter.

- **Evaluation** — In this step the output of the data mining process (previous step) is evaluated. The performance of the model is measured and the results are evaluated from the point of view of the objective that was initially defined.

- **Knowledge Presentation** — Presentation of the results to the user. This presentation depends of the desire output. It can be a visual presentation like data charts or a tables or it can be another data set that can be used as the input for a new data mining process.

Despite presenting a well-defined order, it doesn't necessarily mean that to achieve a good result we are only required to execute only once each one of these steps. In most cases a good result only appear after a few iterations of the whole process [12].

2.2.3 Learning Types

When building a model for an anomaly detection process, it is required to know if the data that is being used in the algorithm to construct the model contains examples that are labeled. In other words, if the value of the objective variable is known [9]. The objective variable is the variable that, in the case of anomaly detection data mining, says to which class a given data instance belong to, normal or anomalous. This is also known as the label. The learning type differs regarding the labelling of the input data. The existent learning types are:

- **Supervised** — The data used to construct the model has labels indicating whether examples are normal or abnormal. This type of learning is very useful for prediction models because it contains information about both classes.

- **Semi-Supervised** — The data used to construct the model contains both labelled and unlabelled examples. Models using this type of learning usually try to define the labelled class properties. Any instances that doesn't fit in those properties is label in the opposite class.

- **Unsupervised** — The data available is not labelled. One example of this type of learning is clustering, which groups the examples in the data according to their similarity.

Sometimes the labelling process is made by hand of an analyst. This is a tiring job and due to this, sometimes the amount of labelled data is very tiny. In some cases there might be bad labelling of the data [13].

2.2.4 Model Evaluation

One of the most important steps when building an anomaly detection model, is evaluating its performance. In the case of anomaly detection, this process consists in extracting indicators which allow to compare one model detection output with te original data. The models output can be of two types:

- **Labels** — for each instance of data, the models outputs a positive or negative label (anomalous or normal).
- **Scores** — for each instance of data the model outputs a value, created by a scoring technique, that represents the *degree* of anomaly also known as *anomaly score*. This allows the results to be ranked and give priority anomalies with higher scores. It is also possible to convert an output *score* into a label by using a threshold.

2.2.4.1 Evaluation Metrics

In anomaly detection, each data instance data can only belong to one of two classes: positive or negative class. The anomolous data is usually represented by the positive class because the objective of the detection process is detect anomolous data. Therefore, the negative class data is the normal data. This notion allows to usage several indicator to evaluate the performance of a given model. Comparing the models output class labels with the original labels 4 indicators can be obtained:

- **True Positives** *(TP)* — Amount of positive class instances classified as positive (anomalous instance labelled as anomalous);
- **True Negatives** *(TN)* — Amount of negative class instances classified as negative (normal instance labelled as normal);
- **False Positives** *(FP)*— Amount of negative class instances classified as positive (normal instance labelled as anomalous);
- **False Negatives** *(FN)*— Amount of positive class instances classified as negative (anomalous instance labelled as normal);

This allows the calculation of more descriptive measures like the *recall* and *precision*, which are the most popular performance indicators for binary classification. *Recall* is the fraction of the original positive class instances that were classified as positive. *Precision* is the fraction of predicted positive class instancess that were correctly classified.

$$precision = \frac{TP}{TP+FP} \tag{2.1}$$

$$recall = \frac{TP}{TP+FN} \tag{2.2}$$

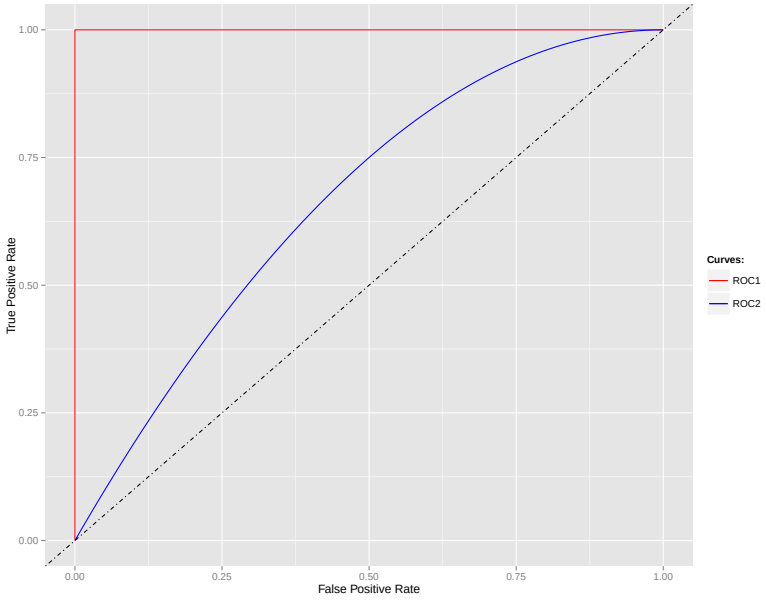

Figure 2.4: 2 models ROC curve Comparison

2.2.4.2 ROC

Another interesting approach to describe the model performance is the ROC curve. To plot a ROC curve it is required for the model to output a probability or a ranking of the examples in terms of some scoring function relative to the positive class [12]. This ranking, or probability, describes how likely it is for an instance to belong to the positive class. This curve plots the relation between the True Positive Rate (TPR), or recall, and the False Positive Rate (FPR) when the boundary of decision changes. The boundary of decision is the threshold used to transform the ranking or probability into a class label. The method to draw the curve is then to make the decision boundary to go from the minimum value to the maximum value int the scoring function, and in each step measure the TPR and FPR.

$$truepositiverate = \frac{TP}{P} = \frac{TP}{TP+FN} = recall \tag{2.3}$$

$$falsepositiverate = \frac{FP}{N} = \frac{FP}{TN+FP} \tag{2.4}$$

Figure 2.4 displays the comparison between two ROC curves. Both curves are drawn above the dotted line $(X = Y)$ which means that the TPR is always bigger than the FPR for every value of the decision boundary. This also means that they are both good models to classify the data. This ROC comparison also displays the perfect model which is represented by the curve *ROC1*. In the perfect model, all the positive instances have are selected by the model before any negative instance is incorrectly selected.

2.2.4.3 Model Validation

Model validation techniques focus on repeating the model construction and validation stages in different testing and training data and in each experiment, to improve the reliability of the estimated values of the performance measures. There are several methods to make the selection of the data used in the train and test stages like the holdout method, k-fold cross-validation, bootstrap and the Monte Carlo experiment. Here, we only describe two of these methods: the k-fold cross-validation and the Monte Carlo experiment.

In the k-fold cross-validation method, the data set is randomly divided into k subsets of the same size with similar initial characteristics (e.g. same class distribution as the original data set). These subsets are also known as *folds*. At each interaction, nine *folds* are used for training and, the remaining *fold* is used for testing. The same fold is never used for training and testing at the same time. The method ends when every *fold* was used for testing [12]. When k is equal to the size of the data set N, this method is called "leave-one-out" cross-validation.

In the montecarlo experiment, the original data set is partitioned M times into disjoint training and testing subsets. The training and testing data sets are fractions of size n_t and n_v, respectively, of the original data set. The main feature of this validation method is that, despite the training and testing data sets being randomly generated, they respect the temporal order of the data. Therefore the training data set is generated from randomly selecting data that is on an earlier time interval than the data used for the generation of the testing data set. This method is especially important for validating time series forecasting models [14].

2.3 Data Mining Techniques for Anomaly Detection

In this section, some of the existing techniques that use data mining processes for anomaly detection will be presented.

2.3.1 Clustering based techniques

Clustering is a process that has the objective of partitioning the data into subgroups of instances that have similar characteristics. These groups are called clusters. Anomaly detection using clusters is made by applying a clustering algorithm to the data and afterwards, the classification of the data is made by one of the following principles:

- **Normal data instances belong to a cluster and abnormal instances don't** — In this case the clustering algorithm shouldn't force every instance to belong to a cluster.
- **Normal data instances are closer to the centroid of the corresponding cluster and abnormal instances are far from it** — The cluster centroid is the graphical centre of the cluster.
- **Normal data instances belong to denser clusters and abnormal instances are in less denser clusters** — In this case it is requires to measure the density of each cluster of data.

To define the density value where a cluster belongs to each one of the classes it is required to define a threshold values;

These techniques don't require the data to be labelled, therefore they are of the type unsupervised learning. They also work with several types of data. Although the anomaly detection is fast and simple after the cluster have been achieved, the clustering process may be slow and computational expensive. The performance of the anomaly detection process depends mostly on the clustering process [9].

2.3.2 Nearest neighbour based techniques

Nearest neighbours techniques are based on measures of the distance (or similarity) between data instances. The distance calculation is made by a relation between the attributes of the data in both instances. Anomaly detection approaches based on nearest neighbour can be of two types:

- **K^{th} nearest neighbour techniques** — This techniques starts by giving each data instance an anomaly score that is obtained by evaluation of the distance to its k^{th} neighbours. Afterwards the most usual way to separate anomalous instances from normal instances is by applying a threshold to the anomaly scores calculated previously.
- **Relative density techniques** — In this technique each instance is evaluated by measuring the density of its neighbourhood. The neighbourhood density is measured by the amount of instances that are within a distance range that is previously defined. It is expected for anomalous data instances to have less dense neighbourhoods and normal data instances to have more dense neighbourhoods.

These techniques can use unsupervised learning, which makes the model construction only guided by the rest of the data features non regarding the class label. In this case, if the normal instances don't have enough neighbours and the anomalous instances do, the model will fail to correctly label the anomalous instances. Since the key aspect of this technique is the distance function, it allows the adaption of the model to other type of data only requiring to change the way the distance is calculated. This is also the main downside, because bad distances calculation methods will result in poor performance [9].

2.3.3 Statistical Techniques

In statistics techniques the anomaly detection is done by evaluating the statistical probability of a given instance to be anomalous. Therefore the first step in these techniques is the construction of a statistical model, usually for normal behaviour, that fits the given data. The probability of an instance to be anomalous is then derived from the model. In the case that the model represents the probability of normal behaviour, when a data instance is evaluated in the model, the ones that are assigned high probability are considered normal and the ones with low probability are considered anomalous. The probability construction models can be of the type:

- **Parametric** — Statistical models of this type assume its probabilistic distribution is defined a priori, and parameters which are derived from the given data. An example of a parametric probability distribution is the Gaussian distribution that is defined by parameters of mean (μ) and standard deviation (σ).

- **Non-parametric** — These models are entirely based on the data without having a probability distribution fixed beforehand. An example of model of this type are the histogram based models, in which the histogram is built using the given data.

These techniques outputs scores associated with a confidence interval. The downside of parametric technique is that works under the assumption that the probabilities are distributed under a specific way which is not true in most cases [9].

2.3.4 Classification Techniques

Anomaly detection using classification techniques requires the construction of a classifier. A classifier is a model constructed under supervised, or semi-supervised, learning algorithm. It represents patterns in the datathat can later be used to classify new data instances as normal or anomalous. There are several classification techniques. The main difference between them are the classifier type (model) that each one builds and use. Some of these approaches are:

- **Neural Networks** — In this approach during the training stage, a neural network is trained with data from the normal class allowing the model to capture the normal data characteristics. Afterwards, data instances are used as input of the network. If the network accepts that instance then it will be classified as normal, if not it will be classified as anomalous.

- **Bayes networks** — In this approach the training stage is used to obtain the prior probabilities for the atributes of the data and class distribution in each class. In the testing stage, the classification is made by calculating the posterior probability (the Bayes theorem) of every class in each data instance. The class with higher probability for that instance, represents the class which that instance is going to be labelled with.

- **SVM** — In this approach the training stage is used to delimit the region (boundary) where data is classified as normal. In the testing stage, the instances are compared to that region, if they fall in the delimited region they are classified as normal, if not, as anomalous.

- **Rules** – In this approach, the training stage is used to create a group of rules (e.g. if-else conditions) that characterize the normal behaviour of the data. In the testing stage, every instance is evaluated with those rules, if they pass every condition they are classified as normal, if not, as anomalous.

These techniques use very powerful and flexible algorithms, because it is only required to apply the classifier to each instance. The downside is that supervised learning requires a very effective labelling processs. If any mistakes were made, they will later result in a worse performance

of the classifier. The output of this technique is in most of the cases a class label, which his not suitable for scoring [9].

2.3.5 Classification with Unbalanced Datasets

An unbalanced data set is a data set where the number of examples of the existing classes is not balanced. This means that the amount of instances belonging to one of the classes is much more abundant than the other(s). In the case of anomaly detection, it is quite usual for the amount of anomaly instances to be quite smaller than the amount of normal ones. This is a problem because it will make the anomaly detection algorithm to identify easily patterns regarding the normal data but not for the anomalies. A solution to counter this problem is to change the model construction process in order to generate conditions to deal with the unbalanced classes. However this solution is not one of the most common because it requires greater research and changes on the standard model construction algorithm, instead of using off-the-shelf algorithm [15]. Therefore some optional solutions that use off-the-shelf algorithms are:

- **Undersample the major class** — before constructing the model, remove some instances of the major class (with sampling techniques). This might result in the decrease of information for the undersample class data in the resulting model.
- **Oversample the minor class** — before constructing the model, repeat some minor class data instances. This will result in some repetitive information being used in the model construction algorithm.

Some additional solutions are based on feature engineering in order to enable the algorithm to find patterns concerning the minority class [16].

2.4 Data Mining on Fraud Detection

Most works on data mining done in telecommunications fraud detection have the objective of detecting or preventing the methods of superimposed fraud [8, 13] and subscription, [1, 17] because these are the fraud types that have done more damage to the telecommunications industry.

2.4.1 Data Format

Projects done in fraud telecommunications aren't always easy to begin, because companies can't allow direct access to data of their clients, due to agreements between carrier and the clients. The type of data that is most commonly avaiable to researchers are the CDR, call details records (or call data records)[6, 2, 1, 5, 13, 17, 18]. These records have a log like format and they are created every time the client finishes using a service of the operator. The format of the records may change from company to company but the most common attributes are:

- Caller and called Identification Number;

- Date and time;
- Type of Service (Voice Call, SMS, etc...) ;
- Duration;
- Network access point identifiers;
- Others;

This data doesn't actually compromise the identification of the users or the content that was traded in the communication, therefore it doesn't violate any privacy agreement. If some logs do compromise the true identification of the clients, the data can be anonymized by the company before being released.

2.4.2 User Profiling

In most fraud detection studies, the authors choose techniques of user profiling in order to describe the behaviour of the users [19, 6, 5, 2]. User profiling is done by separating the data that has anything to do with a user and develop a model that represent its behaviour. This enables the extraction of additional information from the data used, and even add some additional information to the data set, like new attributes, before doing the model for anomaly detection.

The user models can be optimized by dynamically adapting user profiles [8]. This means that the user model must be updated every time the user makes an interaction with the system. Users don't have a constant behaviour so the models that define their behaviour should not be static.

Profiling requires a lot of information about the users in order to build the most representative behaviour model. CDRs are very useful to create behaviour models. However to make a more realistic representation of the user, more information is required, which is rather difficult to obtain due to confidentiality agreements.

2.4.3 Rule based applications

T. Fawcett and F. Provost [8], presented a method for superimposed fraud detection based on rules. They used a group of adaptive rules to obtain indicators of fraudulent behaviour from the data. These indicators were used for the construction of monitors. These monitors defined fraudulent behaviour profiles and are used as input for the analysis model, also known as *detector* [19]. The *detector* combines the output of the monitors with the data generated along the day by a user, and in the case of fraudulent activity, outputs an alarm.

S. Augustin, C. Gaiber, J. Knauer, M. Massoth, K. Piejko, D. Rihm and T. Wiens developed a system for fraud detection in new generation networks[18] . The system is composed by two modules: one for interpretation of data and other for detection of fraud. In the module of interpretation, the CDR's data is used as an input, and the module analyzes and convert that data into CDR objects, which will be used to as input for the classification module. In the classification model,

the CDR objects will be subject to a series of filter analysis. The filters are actually rule based classification models that were obtained previously from real fraudulent data. If a CDR object doesn't respect the rules in those filters an alarm is generated. As a result rate of false-positive 4%.

2.4.4 Neural networks Applications

In the system for fraud detection built by the M. Taniguchi, M, Haft, J, Hollmén and V. Tresp [20], three anomaly detection techniques are used: supervised learning neural networks, parametric statistcal techniques and Bayesian networks. The supervised neural network is used to construct a discriminative function between the fraud and normal classes using only summary statistics. In the parametric statistical technique, the density estimation is generated using a Gaussian mixture model, and it is used to model the past behaviour of every user in order to detect any abnormal change from the past behaviour. The Bayesian networks are used to define the probabilistic models of a particular user and different fraud scenarios, enabling the detection of fraud given a user's behaviour. In the tests done by the authors, the system had a precision of 100% and a recall of 85%.

2.4.5 Other Applications

The system developed by H. Farvaresh and Mohammad M.Sepehri [17] for subscription fraud was composed by 3 stages: pre-processing, clustering and classification. In the pre-processing stage, the raw data was cleaned, prepared and the transformed using principal components analysis (PCA). In the clustering stage, the data was clustered into homogeneous clusters by the two clustering algorithms *SOM* and *k-means*. If the resulting clusters are very specific, very different from other clusters or outlier clusters (one instance cluster), then they are discarded and they are not used in the classification stage. This stage was also used to insert some new features to the data in order to keep the properties of clusters generated previously. In the last stage, the classification stage, the instances are classified using the classifiers SVM, neural networks and decision trees. The classifitaion is made over different ensembles: *bagging*, *boosting*, *stacking*, *voting* and *standard*. The result final result is obtained by the ensemble that offered the best performance.

A very interesting work was done by Kenneth C. Cox, Stephen G. Eick and Graham J. Will [21]. They experimented visual techniques data mining in order to detect anomalies. They claim that these techniques offer one great advantage over learning algorithms, because the systems for the human visualization have better dynamism and adaptability therefore they are better able to cope with the evolution on fraudulent behaviour. These methodology is based in the quality of the graphical presentation of the data on human visual interfaces in order to facilitate the visual fraud detection.

Chapter 3

Naive Bayes

3.1 Algorithm

Naive Bayes is a supervised learning classification algorithm that applies the Bayes's theorem with an assumption of independence between the data features [22]. This assumption is what make him naive. The Bayes's theorem in the classification scenario can be described as

$$P(\theta \mid x_1, x_2, x_3, ..., x_n) = \frac{P(\theta)P(x_1, x_2, x_3, ..., x_n \mid \theta)}{P(x_1, x_2, x_3, ..., x_n)} \tag{3.1}$$

where $x_1, x_2, x_3, ..., x_n = X$ is the data instance (group of features of that instance), θ is the class (positive or negative), $P(\theta \mid x_1, x_2, x_3, ..., x_n)$ is the posterior probability or the probability of a given instance X to belong to the class θ, $P(\theta)$ is the prior class probability distribution, $P(x_1, x_2, x_3, ..., x_n \mid \theta)$ is the prior attribute probability given the class θ, and $P(x_1, x_2, x_3, ..., x_n)$ is the evidence or instance probability.

The *Naive* characteristic of this algorithm is visible on the numerator. Before applying the independence property, the numerator can be re-written using the conditional chain rule:

$$\begin{aligned} P(\theta)P(x_1, x_2, x_3, ..., x_n \mid \theta) &= P(\theta)P(x_1 \mid \theta)P(x_2, x_3, ..., x_n \mid \theta, x_1) \\ &= P(\theta)P(x_1 \mid \theta)P(x_2 \mid \theta, x_1)P(x_3, ..., x_n \mid \theta, x_1, x_2) \end{aligned} \tag{3.2}$$

After all the features have been separated, the independence between features property can be applied and the equation can be summarized as:

$$\begin{aligned} P(\theta)P(x_1, x_2, x_3, ..., x_n \mid \theta) &= P(\theta)P(x_1 \mid \theta)P(x_2 \mid \theta)P(x_3 \mid \theta)...P(x_n \mid \theta) \\ &= P(\theta)\prod_{i=1}^{n} P(x_i \mid \theta) \end{aligned} \tag{3.3}$$

Prior attribute probabilities are calculated by the following equation:

$$P(x_t \mid \theta) = \frac{N_{t,x_t,\theta}}{\sum_{j \in t} N_{t,j,\theta}} \tag{3.4}$$

where $N_{t,x_t,\theta}$ is the amount of instances with attribute t of value x_t belonging to class θ [23]. The denominator ($P(x_1,x_2,x_3,...,x_n)$), is constant in each instance $(x_1,x_2,x_3,...,x_n)$ and results from the sum of the numerator value for all possible classes.

$$P(x_1,x_2,x_3,...,x_n) = \sum_{j=1}^{m} \{P(\theta_j) \prod_{i=1}^{n} P(x_i \mid \theta_j)\} \tag{3.5}$$

Finally, the initial Naive Bayes equation, can be re-written as

$$P(\theta \mid x_1,x_2,x_3,...,x_n) = \frac{P(\theta) \prod_1^n P(x_i \mid \theta)}{\sum_{j=1}^{m} \{P(\theta_j) \prod_{i=1}^{n} P(x_i \mid \theta_j))\}} \tag{3.6}$$

3.1.1 Naive Bayes for Classification

If the objective of the Naive Bayes algorithm is to simply classify instances with a label without regard for the class probability (or ranking) for each instance, then the classification process doesn't require the value of the denominator ($P(x_1,x_2,x_3,...,x_n)$) since it is a constant for each instance. The classification process only requires the result of the numerator, $P(\theta) \prod_1^n P(x_i \mid \theta)$. This characteristic allows the use of Maximum A Posteriori (MAP) value to estimate the values $P(\theta)$ and $\prod_1^n P(x_i \mid \theta)$ to determine which class θ_{pred} does a given instance $(x_1,x_2,x_3,...,x_n)$ achieves its maximum value [22]. The resulting classification rule is

$$\theta_{pred} = argmax_\theta P(\theta) \prod_{i=1}^{n} P(x_i \mid \theta) \tag{3.7}$$

3.1.2 LaPlace Correction

When constructing the Naive Bayes model, namely calculating the class prior probability ($P(\theta)$), there might be some cases where the data does not offer the necessary information regarding a given feature (x_t) for a value (A) in one of the classes (θ_r), resulting in a prior attribute probability of value zero. This will later result in a posterior probability for an instance with feature $x_t = A$ to be zero for the class θ_r. This results in a model that will never classify an instance with that particular characteristic ($x_t = A$) as belonging to the class θ_r even if the rest of the features strongly point for that instance to belong θ_r. To avoid this effect, it is usual to use a method called *Laplace Correction* in order to avoid absense of information in the data regarding one of the classes. This method changes the initial value of the count for instances with that characteristic in a given class

[24]. The formula for a the prior attribute probability calculation for a given attribute x_t on a given class θ is

$$P(x_t \mid \theta) = \frac{I + N_{t,x_t,\theta}}{(K * I) + \sum_{j \in t} N_{t,i,\theta}} \tag{3.8}$$

where I is the Laplace Correction value (usually 1) and K is the different values that attribute x_t can have.

3.1.3 Continuous Variables

Some attributes in the data set might have continues values, this will cause a problem in the likelihood probabilities because it is rather impossible to cover all the values in an interval of continuous values. In order to describe these variables there are some procedures that can be made in order to summarize them. One option is to discretize the variable. In this case the continuous variable will be converted to discrete values (e.g. convert continuous intervals to discrete variables) thus becoming compatible with the standard model construction method. Other common option to describe the continuous variables is by making the assumption that the variable values are follow a parametric probability distribution. This requires some intrinsic adaptions of the Naive Bayes model in order to calculate the likelihood probability using the density distribution function. A popular solution is to use a Gaussian-like distribution. In this case, during training it is only required to calculate the mean (μ_θ) and standard deviation (σ_θ) for the feature x_t in each class θ [25]. During model construction, the prior atribute probability is given by:

$$P(x_t \mid \theta) = \frac{1}{\sigma_\theta \sqrt{2\pi}} e^{-(x_t - \mu_\theta)^2 \big/ 2\sigma_\theta^2} \tag{3.9}$$

3.2 Previous adaptations

A previously study presents a correction to the prior attribute probabilities calculation to counter the unbalanced classes effect on the final model [24]. This solution was studied on the application on the Naive Bayes algorithm with objective of text classification. It can actually be applied as a normalization step to the data, enabling the possibility of leaving the Naive Bayes as a off-the-shelf algorithm. In this particular solution, due to the data structure, the prior attribute probability is calculated by the following equation (using the authors notation):

$$P(w \mid c) = \frac{1 + \sum_{d \in D_c} n_{wd}}{k + \sum_{w'} \sum_{d \in D_C} n_{w'd}} \tag{3.10}$$

where n_{wd} is the number of times that word w appear on document d, D_c is the collection of all documents belonging to class c and K is the previous explained LaPlace Correction. The objective

of this author is to normalize the word count in each class in so that the size for each class is the same. To achieve this, the variable n_{wd} is transformed as:

$$\alpha \frac{n_{wd}}{\sum_{w'} \sum_{d \in D_C} n_{w'd}} \tag{3.11}$$

where α is the normalization constant and defines the amount of smoothing across word counts in the dictionary. When $\alpha = 1$ the resulting equation is:

$$P(w \mid c) = \frac{1 + \frac{\sum_{d \in D_c} n_{wd}}{\sum_{w'} \sum_{d \in D_C} n_{w'd}}}{k+1} \tag{3.12}$$

This allows to scale the prior attribute probability in the case of the objective class (or classes) in order boost the posterior probability calculation for word with those characteristics.

3.3 Proposal

This project proposes the study of two solutions for the anomaly detection system using the Naive Bayes classification algorithm. These proposal are intent to compensate the effect that unbalanced class data generates when constructing the the classification model, and therefore the classification.

For comparison purposes, the standard procedure in the classification task is represented in figure 3.1

3.3.1 Naive Class Distribution

This solution consists mainly in changing the properties of the model generated by the Naive Bayes algorithm, more precisely the prior class probability (($P(\theta)$)) of the model. This approach is the result of analysis of the effect of undersampling the data in the major class before constructing the model. Undersampling the data before applying the Naive Bayes algorithm can result in a better class distribution, which affects the prior class probability ((($P(\theta)$))) of the Naive Bayes model. However, it can degrade que information in the prior attribute probability (($P(\theta \mid X)$)) of the model, which is the result of removing intances by undersampling data. This process on the other hand is meant to balanced the class probability, and therefore the prior class probability without compromising the prior atribute probability. The assumption made in this solution was: *condition the prior class probability without compromising the prior attribute probability, so that the class distribution do not affect the classification*. In order to produce this change, the model construction process must have a variable which describes the prior class distribution. In this case the variable used is the *Prior Percentage Distribution* (*PPD*). (*PPD*) is the value to be assigned to the negative class prior probability ($P(\theta_N)$). Therefore it is also possible to extract the value of the positive class prior probability ($P(\theta_P)$). The method is specified as:

$$P(\theta_N) = PPD \tag{3.13}$$

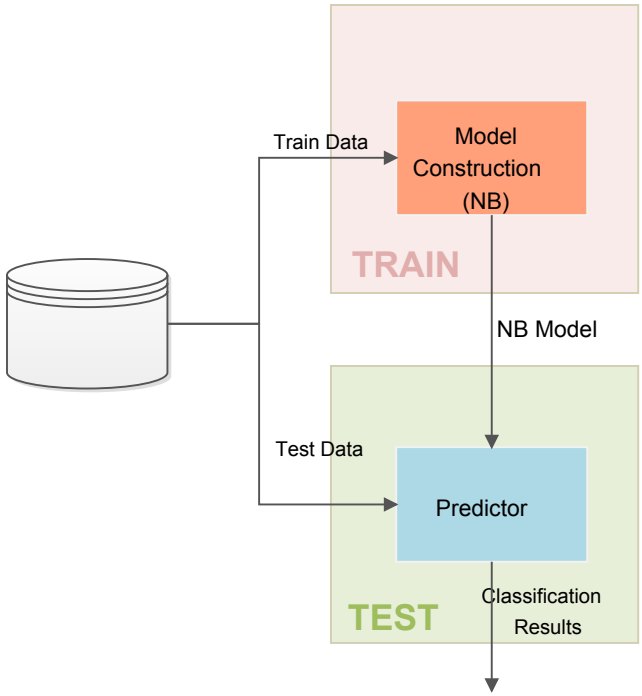

Figure 3.1: Standard Naive Bayes model construction squematics

$$P(\theta_P) = 1 - PPD = 1 - P(\theta_N) \tag{3.14}$$

An representative illustration of the process is presented on figure 3.2.

Comparing to the standard procedure, the main diference is on the training of the model which included *Change Prior Probabilities* module on the training stage. This module changes the values of the prior class probability ($P(\theta)$) according to the previously set *Prior Percentage Distribution* value (*PPD*).

3.3.2 Naive Bayes Margin

This proposal consists on creating additional features to the model generated by the standard algorithm. The *margin values*, *mn* and *mp*, are the key aspect of this proposal and they are built upon the following assumption: *How can the negative class (major) posterior probability be reduced and the positive class (minor) posterior probability be increased in order to get more True Positives without compromising the True Negatives*. This solution has a similiar characteristics to the method in the SVM classification algorithm, where the margin is the distance to the decision boundary. The procedure, is then to study the posterior probabilities for the *True Negatives* and *False Negatives* instances in the classified train data (classified by the model constructed using

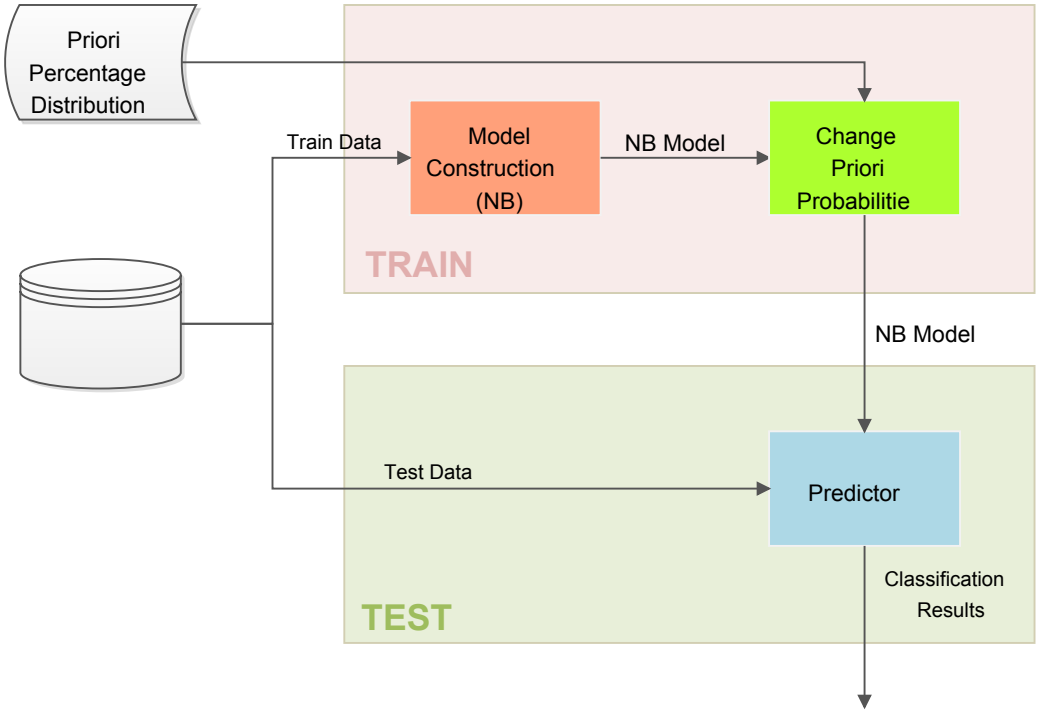

Figure 3.2: Naive Class Distribution model construction squematics

that same train data) in order to achieve two values that can be applied to the probabilities for both classes in order to obtain better Positive Classification. The process used to generate the margin values (*mn* and *mp*) is:

$$mn = \frac{\sum_{i \in TrueNegatives}(P_t(\theta_N \mid i) - 0.5)}{TN} \tag{3.15}$$

$$mp = \frac{\sum_{i \in FalseNegatives}(P_t(\theta_P \mid i) - 0.5)}{FN} \tag{3.16}$$

where $P_t(\theta_N \mid i)$ is the posterior probability of instance i to belong to the negative class (θ_N), TN is the amount of True Negatives, $P_t(\theta_P \mid i)$ is the posterior probability of instance i to belong to the positive class (θ_P) and FN is the amount of False Negatives. Note that this process occurs during the training stage, therefore these are posterior probabilities for the train data. Also note that *mn*, that is the *margin value* generated from the true negative probabilities, is a positive value while *mp*, that is the *margin value* generated from the false negative probabilities, is a negative value.

The generated *margin values* are then applied to the process after the classification process. The application method is represented in the equation:

$$P_m(\theta_N \mid X) = \begin{cases} \frac{P_p(\theta_N|X)-mn}{(P_p(\theta_P|i)-mp)+(P_p(\theta_N|X)-mn)} & (P_p(\theta_N \mid X)-mn) > 0 \\ 0 & (P_p(\theta_N \mid X)-mn) < 0 \end{cases} \tag{3.17}$$

$$P_m(\theta_P \mid X) = \begin{cases} \frac{P_p(\theta_P|X)-mp}{(P_p(\theta_P|X)-mp)+(P_p(\theta_N|X)-mn)} & (P_p(\theta_N \mid X)-mn) > 0 \\ 1 & (P_p(\theta_N \mid X)-mn) < 0 \end{cases} \tag{3.18}$$

where the $P_p(\theta_P, X)$ and $P_p(\theta_N, X)$ are the posterior probabilities generated by classification process with the naive bayes model and $P_m(\theta_P, X)$ and $P_m(\theta_N, X)$ are the new posterior probabilities after applying the margin to the predictions, both cases for intances $X = \{x_1, x_2, ..., x_n\}$. Note that the denominator is only a normalization process that is applied so that posterior probabilities keep the property of $P_m(\theta_P, X) = 1 - P_m(\theta_N, X)$.

The process for the model construction and the new features generation are represented in figure 3.3.

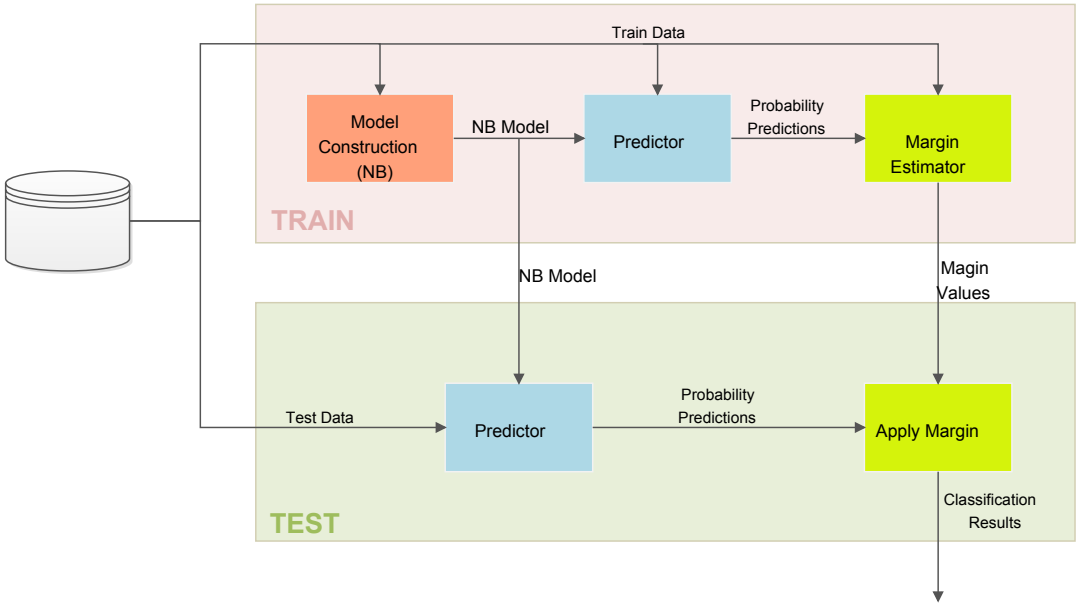

Figure 3.3: Naive Bayes Margin model construction squematics

The main differences for this process with the standard procedure are:

- **Train** — Instead of simply using the training stage to construct the Naive Bayes model, after the construction the resulting model will be used to classify (*Predictor* in the train area of figure 3.3) the same data that is used for its construction (train data). The output of that classification process is in the form of posterior probabilities ($P_t(\theta \mid X)$). In this case, the

probabilities for an instance to belong to the positive class ($P_t(\theta_P \mid X)$) or the negative class ($P_t(\theta_N \mid X)$). Afterwards, the train data and the associated posterior probabilities will feed the *margin estimator*. The *margin estimator* is going to derive two new values (*mn* and *mp*) from that train data and correspondent classification, that will be used in future predictions. These values are denominated as *margin values*.

- **Test** — In the testing stage, the model without the new features is going to be used for the prediction as the typical classification process would do. The output of the prediction process ($P_p(\theta \mid X)$) also needs to be in the form of class probabilities (like the predictor in the training stage). Instead of using these probabilities to label the test data and finish the process by evaluating the results, the posterior probabilities will feed, together with the previous calculated *margin values* into the *apply margin* module. In this module, the posterior probabilities generated by the model ($P_p(\theta \mid X)$) are re-calculated using the designed algorithm and the previously calculated *margin values*. This process will results in new posterior probabilites ($P_m(\theta \mid X)$) that will be used to label the data.

Results of this proposals will be presented in the results section further in this document.

Chapter 4

Prediction of Telecom Anomalies

4.1 Data Understanding

The data set used for this project is composed by two days of CDR (Call Details Records) from a telecom company that must remain anonymous for confidentiality reasons. Each data set has the following features (or attributes):

- **DATE** — date in which the record was created. It is in the format "YYYYMMDD", "YYYY" year, "MM" month and "DD" days. This parameter has only two different values in all examples of the data set used, because there are only two different days.
- **TIME** — time in which the record was created. It is in the format "hhmmss", "hh" hours, "mm" minutes and "ss" seconds.
- **DURATION** — is the amount of time, in seconds, that the service lasted.
- **MSISDN** — anonymized cellphone number of *user1*.
- **OTHER_MSISDN** — anonymized cellphone number of *user2*.
- **CONTRACT_ID** — contract indentification number of *user1*.
- **OTHER_CONTRACT_ID** — contract indentification number of *user2*.
- **START_CELL_ID** — cell identification number for the cell (antenna) where the caller user is accessing.
- **END_CELL_ID** — cell identification number for the cell (antenna) where the called user is accessing.
- **DIRECTION** — describes the direction of the call. It can take two values: "I" for inbound and "O" for outbound. This atribute describes the *user - Caller/Called*. If the value is "I"(inbound) then *user1* is the caller and *user2* is the called user.

- **CALL_TYPE** — identifier of the type of the call, or service, which the record corresponds to. It can take the following values: "MO", "ON", "FI", "SV" and "OT". A precise description of what those values mean was not provided.
- **DESTINATION** — describes the territorial of the call. It can take two values: "I" for international calls and "L" for local calls.
- **VOICEMAIL** — a flag-like varialble that takes the value "Y" for calls who went to voicemail and "N" for calls who are not
- **DROPPED_CALL** — target variable for the anomaly detection system, or Class. It is a flag-like feature that take two values: "Y" for a dropped call (Positive Class) and "N" for a successful call (Negative Class).

An exploratory data analysis was carried out by computing basic statistics and visualizing the data. Several interesting observations were made, including:

- Cells identifiers, **START_CELL_ID** and **END_CELL_ID**, are not available in every instance. There isn't a relation in the call characteristics that justifies the missing value because, in some cases, for calls with the same characteristics there are missing values in both IDs or only one ID or even none of them is missing. This effect could relate to calls which the users are accessing the networks trough cells who don't belong to the carrier infrastructure, therefore they can't be identified.
- The contract identifier of *user2*, **OTHER_CONTRACT_ID**, in most of the cases is not filled in. This could also relate with the cells identifiers effect explained previously, but in this case it is not the infrastructure that belongs to another carrier but the client.
- All international calls (calls with **DESTINATION** equal to "I") have the **CALL_TYPE** feature always equal to "OT", while in the local calls the **CALL_TYPE** feature can take values of "MO", "ON", "FI" and "SV";
- All calls who went to voicemail (**VOICEMAIL** have the value "Y") have the **CALL_TYPE** feature equal to "SV" and all have the same **OTHER_MSISDN** number. This might correspond to the anonymized number of the voicemail center;

There are some important characteristics about this data that can be displayed through some simple statistics. The total amount of records used for this project is of 14,043,142 for the *day1* data set and 13,888,323 for the *day2* data set. The class distribution is visibly unbalanced (figure 4.1) where *Day1* has about 2.87% dropped calls and *Day2* about 2.95%.

There are some important characteristics about this data that are required to be presented because they associate the present data with data that can actually be generated by a true telecommunications carrier in a real-world scenario. The first characteristic is presented in figure 4.2 in which the amount of calls that are made per hour during each of the days are displayed. It is possible

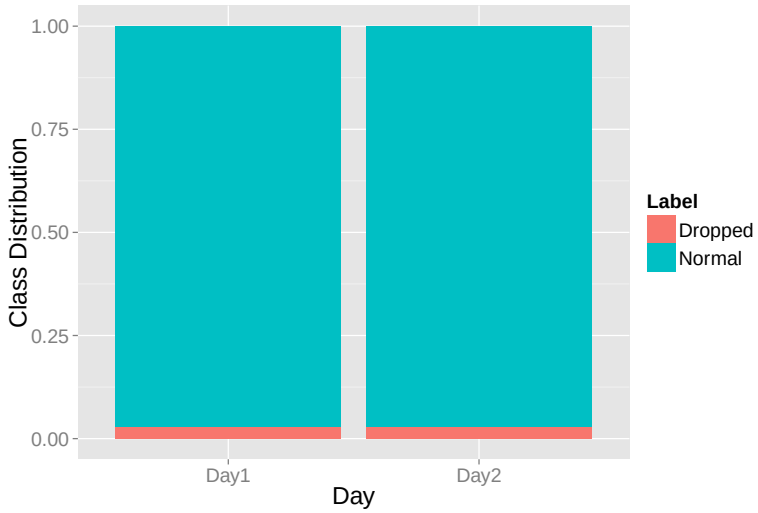

Figure 4.1: Class distribution in each day

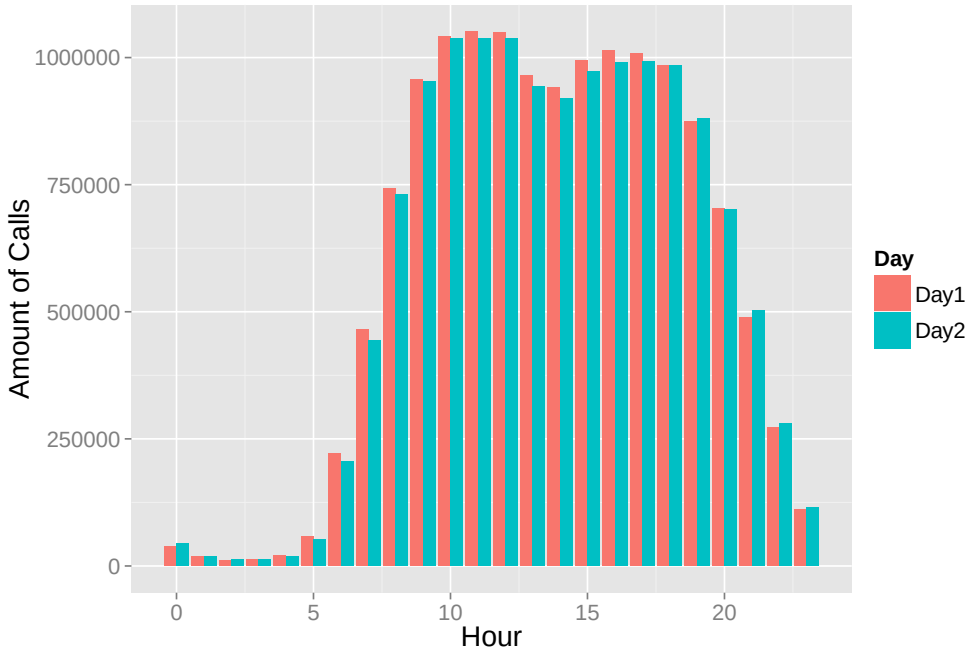

Figure 4.2: Number of calls per hour in each day

to see that the amount of calls that are done in the day time (8am to 9pm) is much larger than the amount of calls done during the night time (10pm to 7am) which is the expected distribution. Another interesting aspect is shown in figures 4.3 and 4.3. These figures, represent the amount of calls done by duration in each of the two days. In this case, it is possible to observe that most of the calls are of short duration (< 100 seconds) which is very typical on a real-world scenario. The call duration presented in figures 4.3 and 4.4 exclude some outlier calls with duration > 5000 sec-

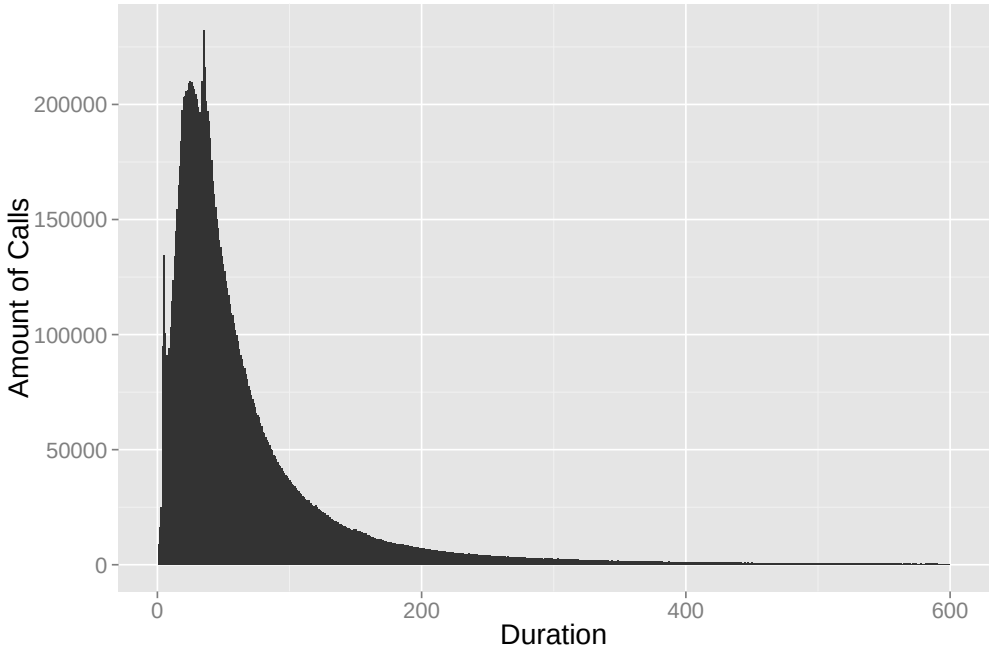

Figure 4.3: Number of calls by duration in day 1

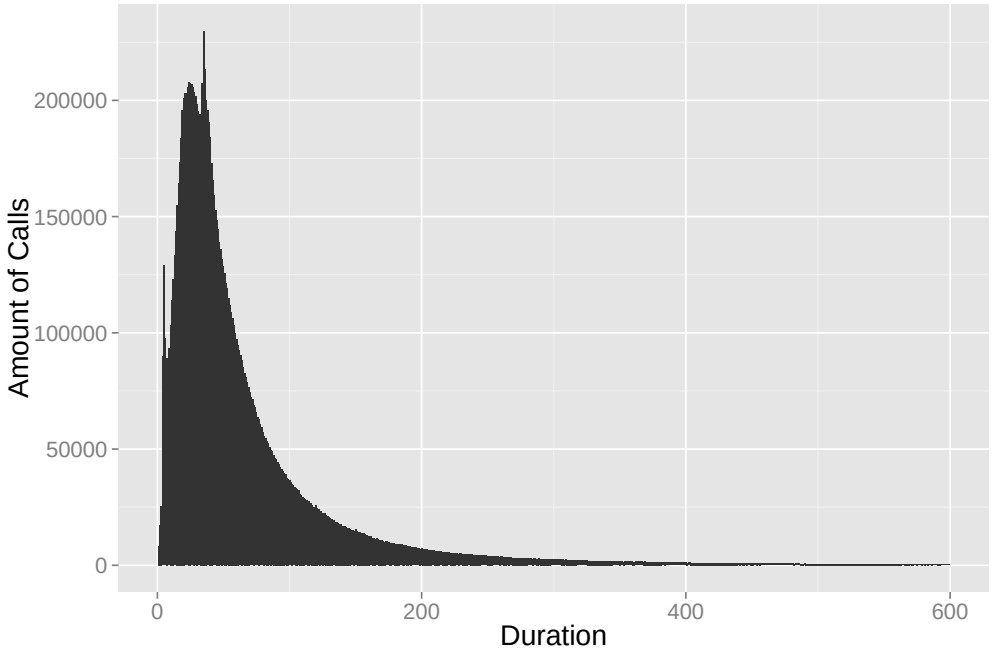

Figure 4.4: Number of calls by duration in day 2

onds. One important property that can be obtained from this figures that is going to help the model construction is that, the call duration presents a Gaussian-like distribution (see section 3.1.3).

4.2 Data Preparation

In attempt to obtain more information concerning the target variable, some extra features were generated and added to the data. These features objective is to include more information about the relation between the call properties and the cell and user which the call belongs to. The features format is designed to be compatible with the naive bayes algorithm. In brief, this feature generation process starts by building a characterization of the cells and users with data up to a certain time t. These characteristics are then appended to the variables that describe calls with date t+1. These new features are also designated as *signatures*, because they add to the data, information regarding the characteristics of the users and cells involved in each particular call.

The process for this signature generation is composed by two essential stages:

- **Data Analysis and Signature Database Generation** — In this stage the data set is processed in order to group data by cell and user id. Afterwards, statistics are made regarding the number of occurrences in the other features (e.g. DROPPED_CALLS, DESTINATION). These statistics are very straightforward and simple , such as the *number of successful international calls in cell with ID J when this is the start cell* or *number of dropped inbound calls in cell with ID J when this is the end cell*. To take into account the differences in behavior of the users at different times of the day, intervals were made in order to group data (e.g. TIME was grouped in 8 hour intervals (0-7, 8-15, 16-23)).

- **Signature Generation** — In this stage the data instances are extended with the corresponding signature features. In each instance the cells and user ids are used to access the information generated and stored in the previous stage. This information is then combined with the call instance properties to generate the new features.

For each cell and user id, two new features are generated. The first describes the association of the call properties with the cell (or user) normal characteristics (i.e. the characteristics associated with calls from the Negative Class) and other that describes the association of the same call properties with the cell or user abnormal characteristics (Positive Class).

The main operator used in the generation of the features is the relation between the attributes in the call and the data regarding the class and the cell (or user) identifier. The operator $P_j(X_i \mid \theta_P)$ and $P_j(X_i \mid \theta_N)$, is defined as the probability for a given instance X_i to belong to the positive θ_P or negative θ_N class in a user or cell with id j. This properties are very similar to prior attribute probabilities in the Naive Bayes algorithm, the main difference is that the data used to generate the probabilities belongs to a specific user or cell.

In the first approach, the feature calculation was based in a independence relation between the main operators in all features. Similar to the independence between features present in the Naive Bayes algorithm. The calculation method for the *Negative Signature* is:

$$NegativeSignature1 = \prod_{i=1}^{n} P_j(X_i \mid \theta_N) \tag{4.1}$$

And the *Positive Signature* equation is

$$PositiveSignature1 = \prod_{i=1}^{n} P_j(X_i \mid \theta_P) \tag{4.2}$$

The second approach result in some side effects that were detected in the first approach. The previous method used the multiplication between attributes, which might result in the low probability attributes to affect the high probability of the others. To counter this effect, the second approach sums all the attributes probabilities. Therefore the low probabilities attributes will not affect the final signature value in such a severe manner. The second approach for the *Negative Signature* equation is:

$$NegativeSignature2 = \sum_{i=1}^{n} P_j(X_i \mid \theta_N) \tag{4.3}$$

And the *Positive Signature* equation is:

$$PositiveSignature2 = \sum_{i=1}^{n} P_j(X_i \mid \theta_P) \tag{4.4}$$

The objective of building two features like this, it to give each call an additional characteristic that allows an easier association with normal or abnormal properties. Another important aspect about this features is that when they are included in the data, the atribute cell or user id that is used to make the relation between the call and the cell, or user, is removed from the data, because they do not have any discriminating power in the Naive Bayes algorithm. In total there are 6 features that are added when the signature is generated. One pair for the start cell id, other for the end cell id and other for contract id. However, in the case of the other contract id there is no signature generated because most of the instances do not have any information regarding the other cell id.

4.3 Methodology

Given that the data used in this project is a stream of calls, it was necessary to construct a testing environment that would behave like the real world application. Therefore, the use of the data to construct the models and then too validate them, had to take into consideration the temporal ordering of the instances. In other words, model and signatures were generated from older data than the data they are used to classify. This rule was followed when training and testing the models, therefore a monte-carlo experimentation routine was used in order to get the training and testing data ordered in temporal order.

Figure 4.5 shows the simulated call classification process. This is achieved by previously training the model and generating the database signatures so that it is only required to apply the signature and classify each call individually. Each call is inserted in this simulation by the Call Arrival input with the temporal order defined by the call timestamp. The *insert signature* module is meant to add the signature features to the call as explained in section 4.2. This module has 3 possible configurations: include signature of type 1 (first approach), include signature of type 2

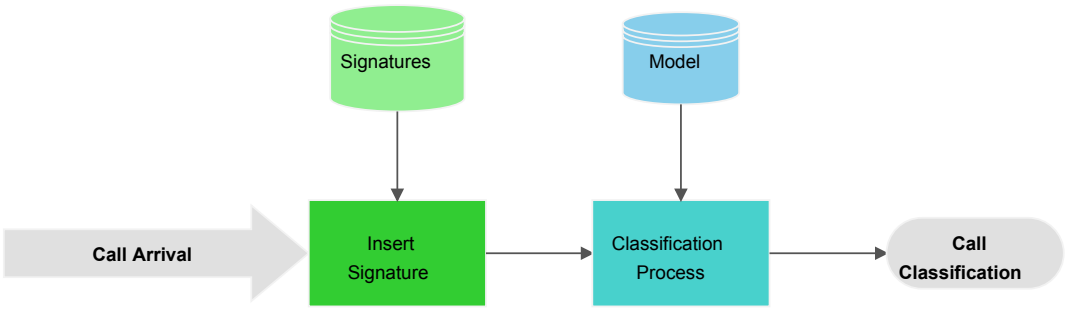

Figure 4.5: Call Classification Simulation Process

(second approach) or don't insert any signature. The *signature database* is constructed using the methods explained in section 4.2 with a data set containing a full day of past records (*day1* of section 4.1), therefore respecting the temporal order rule that was established in the earlier. The *classification process* module uses the previously generated and stored model to classify the data. This allowed to test 4 types of model constructions which are named as: *Normal Naive Bayes*, *Undersampled Naive bayes*, *Prior Naive Bayes* and *Maximal Positive Margin Naive Bayes*.

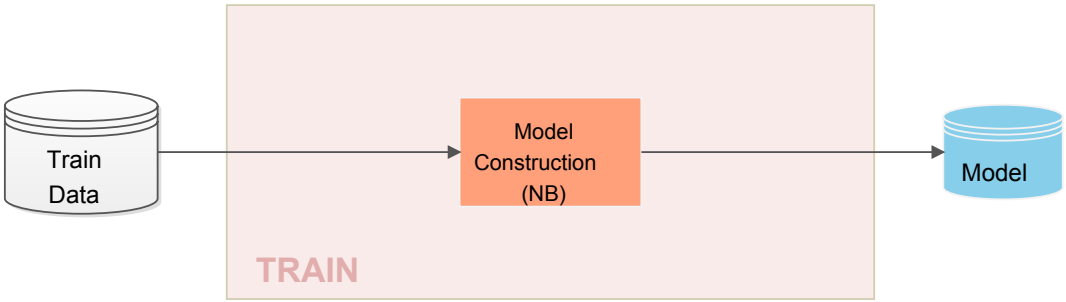

Figure 4.6: Normal Naive Bayes Model

The *Normal Naive Bayes* process, shown on figure 4.6, is the normal Naive Bayes model construction process where the model is obtained simply by applying the traditional Naive Bayes algorithm to the training data.

The *Undersampled Naive Bayes* process is shown on figure 4.7. The training set used for this model construction is obtained by undersampling the negative class (major class) instances according to a pre-defined percentage, without changing the amount of positive class instances. The model is obtained by applying the traditional Naive Bayes to the resulting *undersampled data*. The *undersampling rate* parameter defines the amount of negative class instances that are going to be sampled from the complete dataset. For instance, an *undersampling rate* of 20% in 10,000 negative class instances will result in a 2.000 random sample of those negative class instances.This

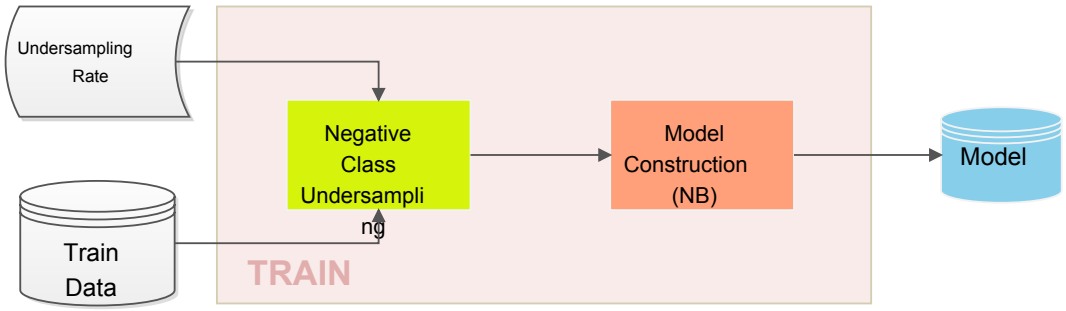

Figure 4.7: Undersample Naive Bayes Model

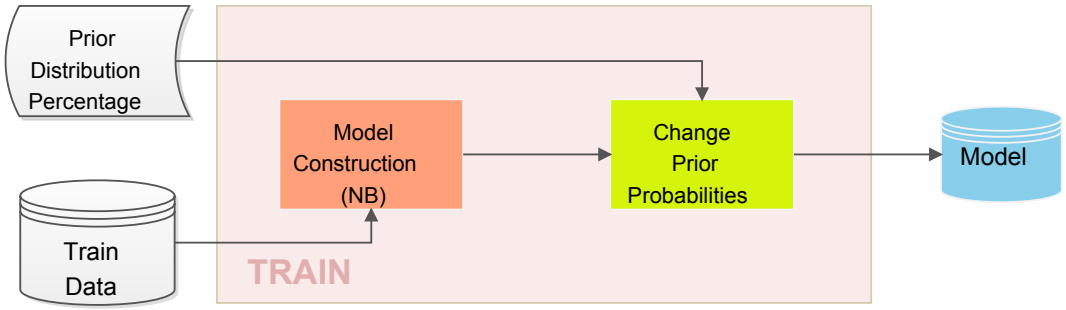

Figure 4.8: Prior Naive Bayes Model (Naive Class Distribution proposal)

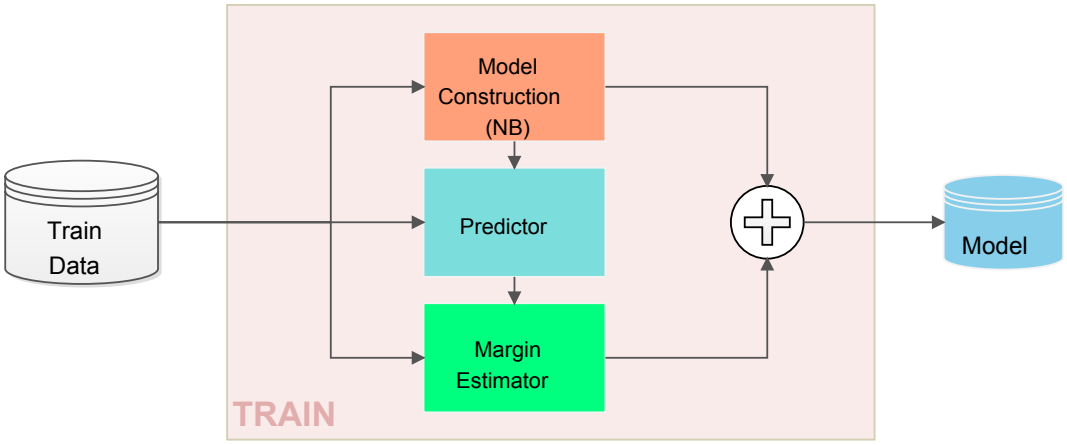

Figure 4.9: Maximal Positive Margin Naive Bayes Model (Naive Bayes Margin proposal)

undersample process outputs the data in the correct time order therefore respecting the previously defined rule.

The *Prior Naive Bayes* process, shown on figure 4.8, is represents the *Naive Class Distribution* proposal. It starts by using the training data to construct the model as the *Normal Naive Bayes* process would. After the model is constructed, it changes the prior class probability in the model with the *prior probability distribution* parameter given. For instance,a *prior probability percentage* of 20% will result in a positive prior class distribution of 0.2 ($P(\theta_{\mathrm{N}}) = 0.2$) and a negative prior class distribution of 0.8 ($P(\theta_{\mathrm{P}}) = 0.8$).

The last process is the *Maximal Positive Margin Naive Bayes* process which is shown in figure 4.9 and represents *Naive Bayes Margin* proposal. This process is described in the proposal section 3.3.2.

All of these models construction processes, with every 3 possible signatures and different configurations (in the case of the models with variable percentage values) were tested using the monte-carlo method for estimating performance (section 2.2.4.3). The monte-carlo experiments configurations used were: training data size and testing data size of 30 and 10 percent of the data ,respectively.The number of repetitions in the model validation was different of 10 repetitions for the models without any predefined parameter (*Normal Naive Bayes* and *Maximal Positive Margin Naive Bayes*) and 5 repetitions for the models with predefined parameters (*Undersample Naive Bayes* and *Prior Naive Bayes*). The reason behind the different number of repetitions in different model validations relies on the total number of interactions that were required to obtain results. In the models with predefined parameters, the validation requires to be done with different parameter values in order to verify the differences that the parameter causes in the classification.

4.4 Results

The results displayed in this section are presented in the order in which the components were studied and integrated. In all the models tested, the performance is measured by analysis of the *recall* and *precision* of the classification process.

4.4.1 Normal Naive Bayes Results

The first stage of this project was to evaluate the performance of the standard Naive Bayes model. This model is expected to have a poor performance due to the characteristics of the data. In the methodology section (4.3) this process is named under *Normal Naive Bayes Model*.

Figures 4.10, 4.11 and 4.12 display the performance indicators of the *Normal Naive Bayes Model* with the several signature types. The *recall* max value reaches around 6.7% for the model using data with signature of type 1 and the *precision* reaches a maximum value around 11% for the model using data with signature of type 2. The perfomance of this models is as expected, very weak. The data unbalanced class characteristic forces more instances being classified as negative, which results in low a *recall* and *precision*. Both values of best performance in *recall* and *precision* in this model were reached using data with the designed aditional features.

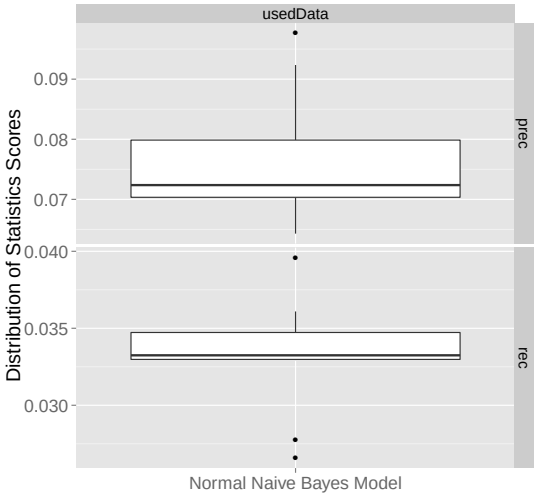

Figure 4.10: Normal Naive Bayes model using data without signature

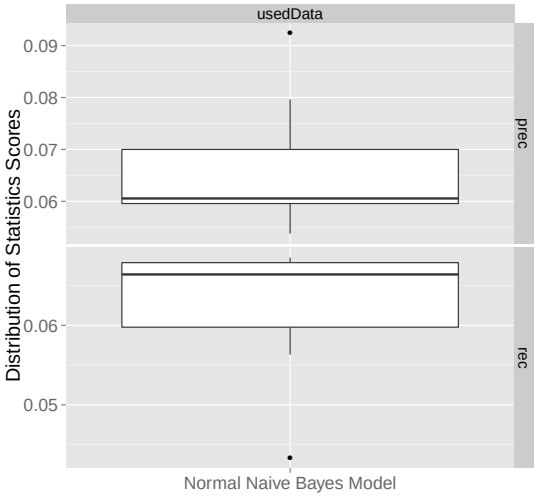

Figure 4.11: Normal Naive Bayes model using data with signature of type 1

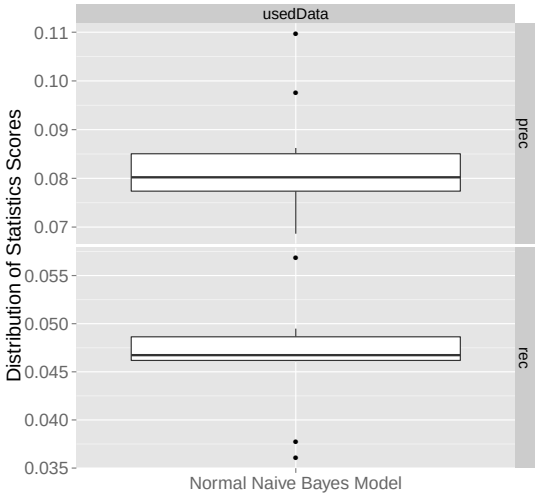

Figure 4.12: Normal Naive Bayes model using data with signature of type 2

4.4.2 Undersampled Naive Bayes Results

The *Undersample Naive Bayes Model* (section 4.3) is the representation of a previously studied solution to reduce the unbalanced data set effects. The hypothesis in evaluating this model is: *investigate what effects does a variable undersampling rate produces in the resulting models performance*.

Figures 4.13, 4.14 and 4.15 display the performance of the *Undersampled Naive Bayes Model* built with the several signature types and the *undersampling rate* parameter ranging from 10% to

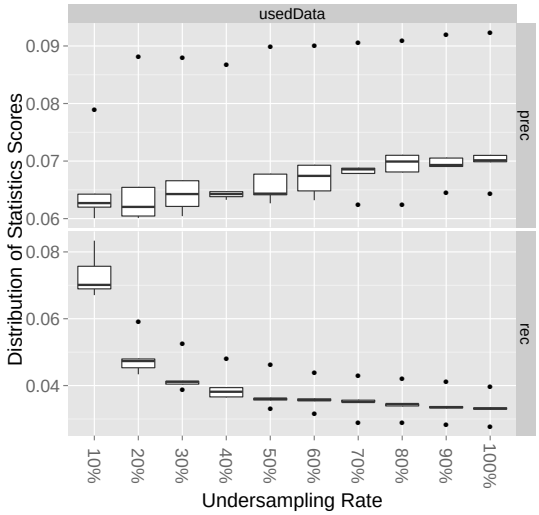

Figure 4.13: Undersampled Naive Bayes model using data without signature

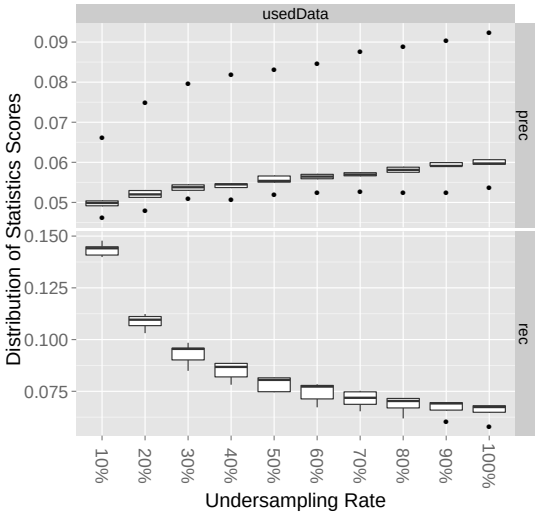

Figure 4.14: Undersampled Naive Bayes model using data with signature of type 1

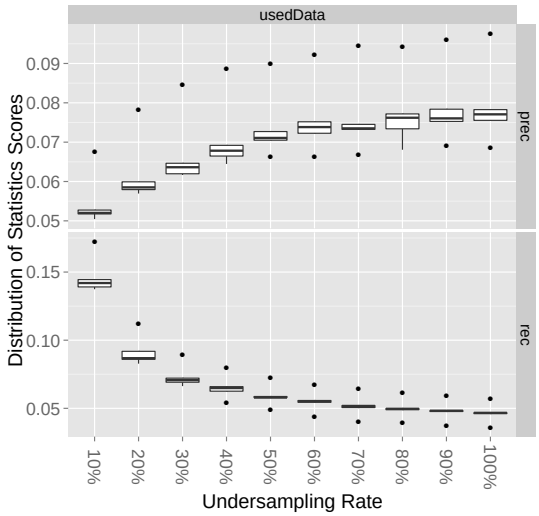

Figure 4.15: Undersampled Naive Bayes model using data with signature of type 2

100% with a 10% step. For an 10% *undersampling rate* the maximum *recall* was 17.5% for the model of data with signature 2, and the maximum *precision* was 7.8% for the model of data without signature. As the *undersampling rate* increases, the *precision* raises and the *recall* drops. This effect is visible in the model with all the signature types. When the *undersampling rate* reaches 100%, the results should be similar to the *Normal Naive Bayes Model*, because no undersampled is made. This is confirmed as the maximum *recall* reaches 6.2% for the model of data with signature 1 and the maximum *precision* reaches 10.5% for the model of data with signature 2.

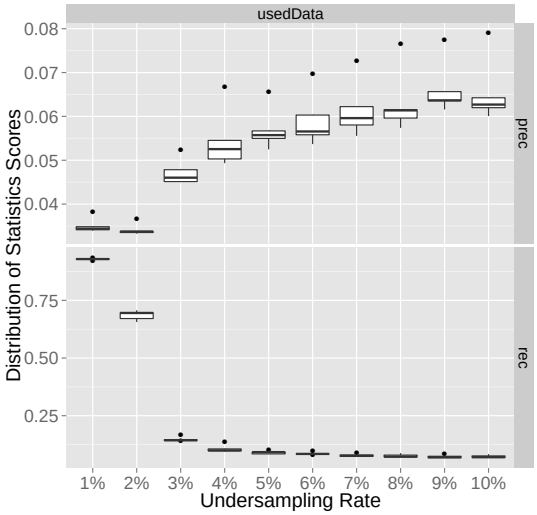

Figure 4.16: Undersampled Naive Bayes model using data without signature (1% to 10% undersampling rate)

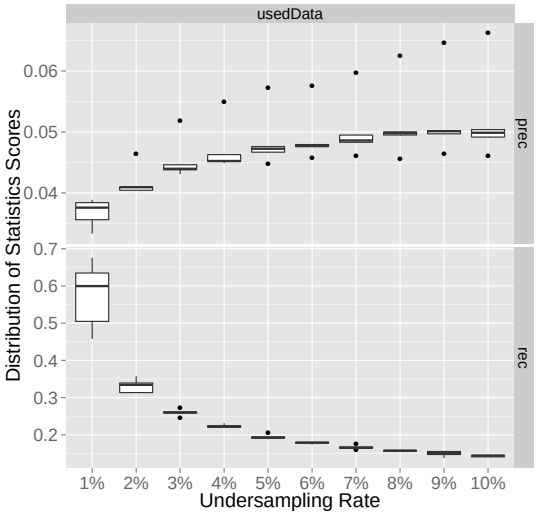

Figure 4.17: Undersampled Naive Bayes model using data with signature of type 1 (1% to 10% undersampling rate)

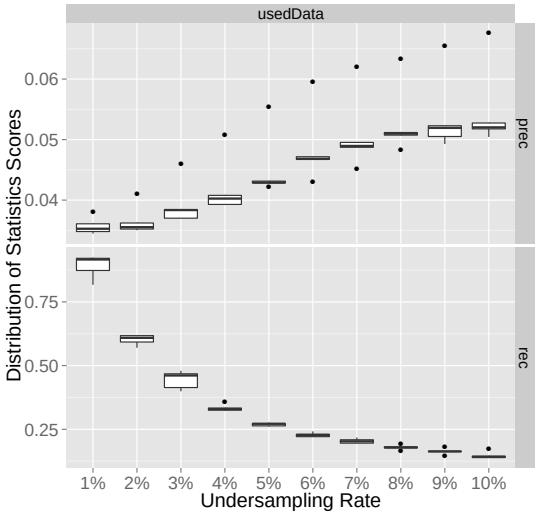

Figure 4.18: Undersampled Naive Bayes model using data with signature of type 2 (1% to 10% undersampling rate)

Figures 4.16, 4.17 and 4.18 display the performance of the *Undersampled Naive Bayes Model* with the several signature types but with *undersampling rate* ranging from 1% to 10% with a 1% step. These results show that for an *undersampling rate* of 1% the *recall* reaches a maximum of around 85% and the *precision* a maximum of 3.8% for the model of data without signature. At 1% *undersampling rate* value, the negative prior class probability is smaller than the positive prior probability in the Naive Bayes model which boosts the Positive classification. The *recall*

has a drop in percentage between the *undersampling rate* of 2% and 3%. This is because, the *undersampling rate* is in the range of values in which the class distribution reaches the even point ($P(\theta_N) \approx P(\theta_P)$).

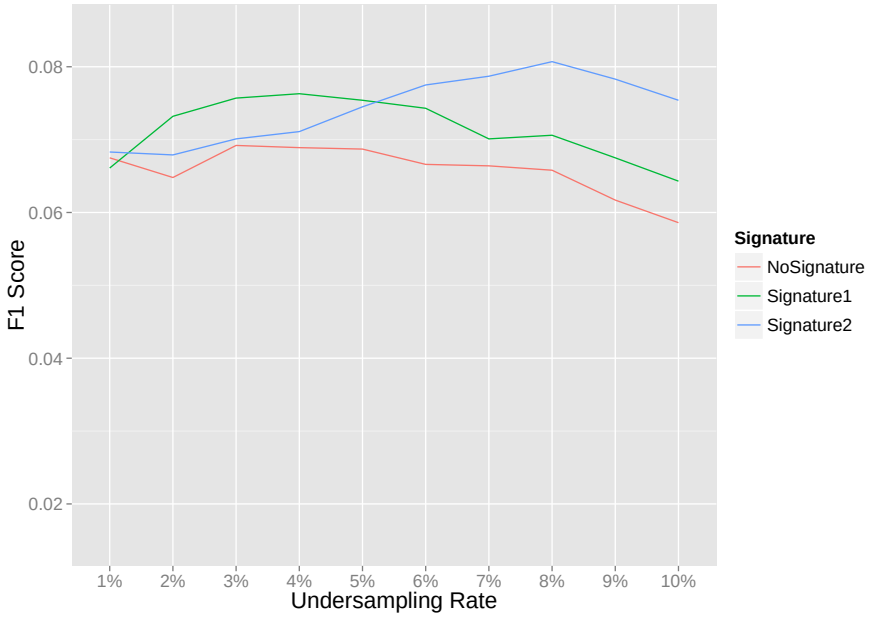

Figure 4.19: Undersampled Naive Bayes model F1 Score (1% to 10% range)

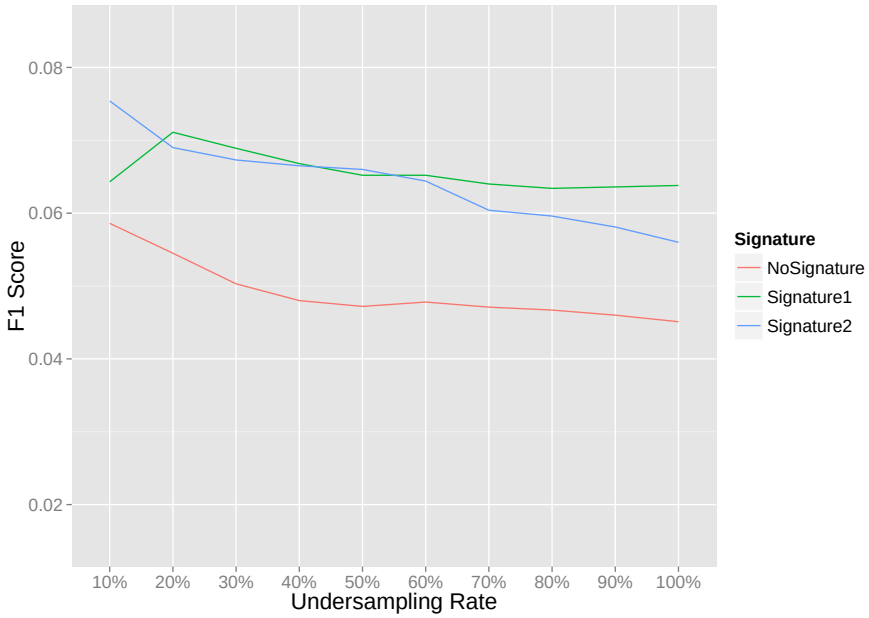

Figure 4.20: Undersampled Naive Bayes model F1 Score (10% to 100% range)

In conclusion, the reduction of the *undersampling rate* produces an increases of the positive

prior class probability ($P(\theta_P)$) in the Naive Bayes model which results in more calls being classified as Positive. Since more calls are classified as positive, the *recall* increases due to an increased amount of True Positives classifications and the *precision* decreases due to an increase of the amount of False Positives classifications. Yet the growth of the *recall* is faster then the decreasing of the *pricision* as shown of figures 4.19 and 4.20.

4.4.3 Prior Naive Bayes Results

The *Prior Naive Bayes Model* (section 4.3) is the representation of the proposal *Naive Bayes Margin*. The first hypothesis in the evaluation of this solution is: *investigate how the changes in the model prior class probabilities improve the ability to detect positive examples*. The second hypothesis is: *investigate the diferences between this solution and the undersampled solution*.

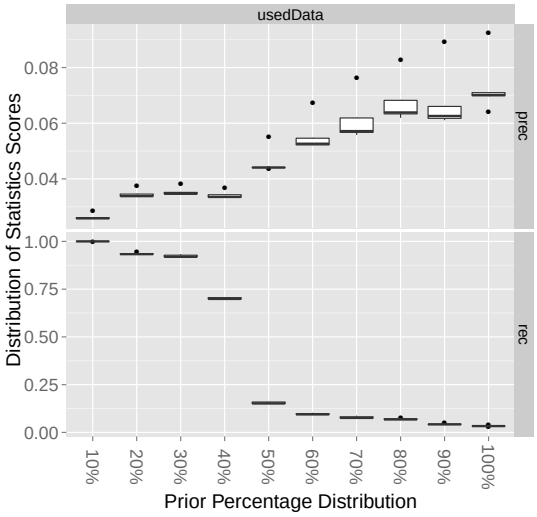

Figure 4.21: Prior Naive Bayes model using data without signature

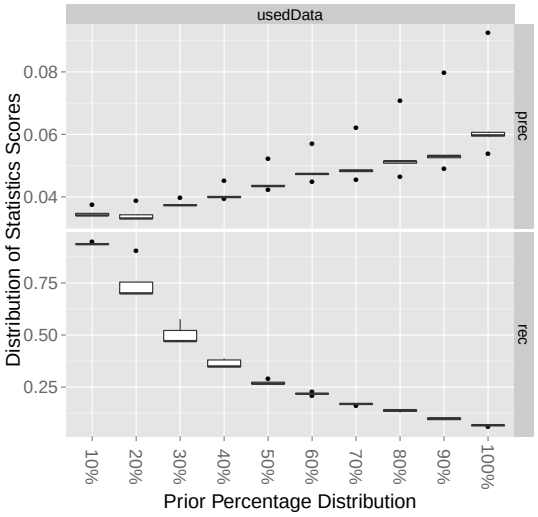

Figure 4.22: Prior Naive Bayes model using data with signature of type 1

Figures 4.21, 4.22 and 4.23 display the performance of the *Prior Naive Bayes Model* with the several signature types and the *Prior Percentage Distribution* (*PPD*) parameter ranging from 10% to 100% with a 10% step. For a *PPD* of 100% the results correspond to the original class distribtion, which is around 98% (section 4.1). This results show an increasing *precision* and a decreasing *recall* as the *PPD* increases. This is the result of the naive bayes model having a prior class distribution ($P(\theta)$) that changes from a major positive class ($P(\theta_P) > P(\theta_N)$) when the *PPD* is below 50% to a major negative class ($P(\theta_N) > P(\theta_P)$) when the *PPD* is above 50%. The first hypothesis is verified when the *PPD* has low values, which results in an increased positive classification due to the increased positive prior class probability in the Naive Bayes model. For a *PPD* of 10% the *recall* reaches a maximum of 100% with a low *precision*. Same as the *Undersampled Naive Bayes Model* the increased positive classification will generate more True Positives and False Positives. The second hypothesis is verified in the prior attribute probability of the Naive

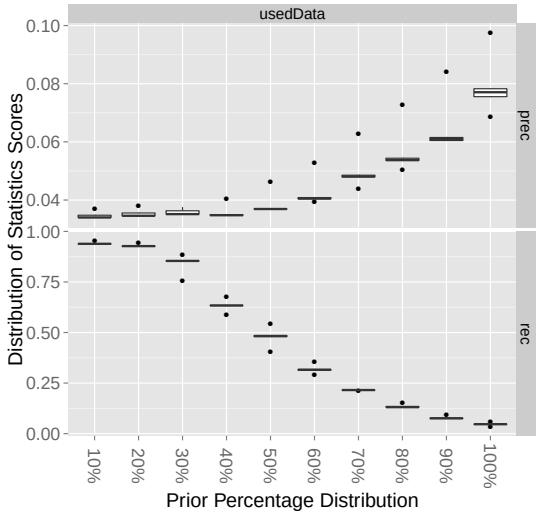

Figure 4.23: Prior Naive Bayes model using data with signature of type 2

Bayes model. In the *Undersampled model*, the *undersampling* task removes instances before constructing the model in order to balance the class distribution. This might remove some essential information regarding the negative class, which later might compromise the classification process. In the case of the *Prior Naive Bayes Model*, no information is removed and the desired prior class distribution is defined manually.

Figure 4.24: Prior Naive Bayes model F1 Score

4.4.4 Maximal Positive Margin Naive Bayes Results

The *Maximal Positive Margin Naive Bayes Model* (section 4.3) is the representation of the *Naive Bayes Margin* proposed solution. The hypothesis in this solution is:*investigate how can the margin between the the negative classification and the decision boundary be decreased in order to obtain te detect more positive examples*.

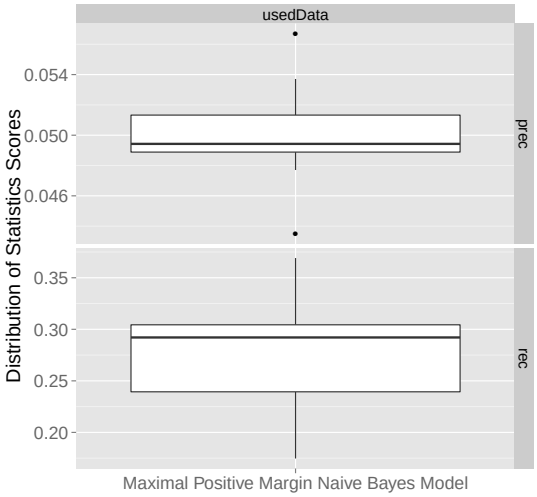

Figure 4.25: Maximal Positive Margin Naive Bayes model using data without signature

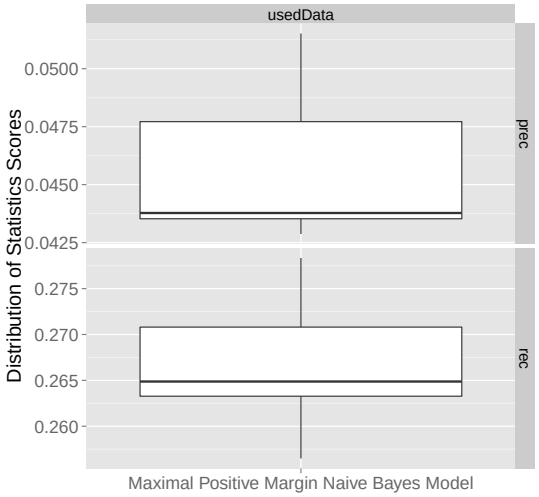

Figure 4.26: Maximal Positive Margin Naive Bayes model using data with signature of type 1

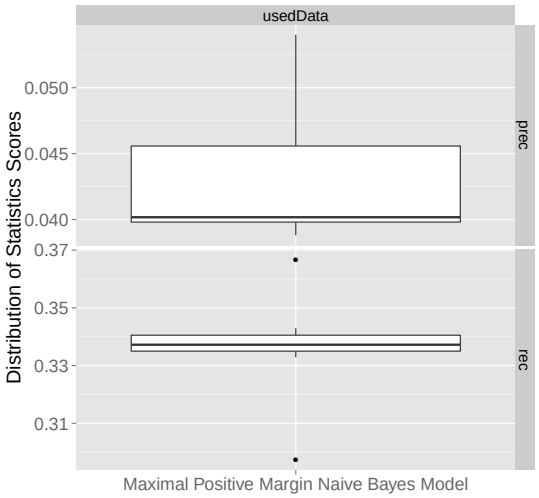

Figure 4.27: Maximal Positive Margin Naive Bayes model using data with signature of type 2

Figures 4.25, 4.26 and 4.27 display the performance of the *Maximal Positive Margin Naive Bayes Model* with the several signature types. In general, the *precision* has worsen value compared to the *Normal Naive Bayes* model, maximun of around, 5% yet the *recall* is higher, maximum of around 36%. This is because some of instances have been moved from the negative class to the positive class, resulting in more True Positives (that justify the improved *recall*) and more False Positives (decreased *precision*). In this solution, after applying the margin values to the model results, the only effect that occurs to the results are some of the classified negative instances (*True Negatives* and *False Negatives*) being converted into positives (*True Positives* and *False Positives*). Table 4.1 displays the amount of instances that were converted to positives and the fraction that those values correspond in the total amount of positive and negative instances. At a first glance it is possible to see that applying this solution to data with all types of signatures, the fraction of true positives (from the amount of positives instances) gained is bigger than the fraction of true negatives (from the amount of negatives instances) lost.

Signature	added TP	added FP	Pos. Fraction (%)	Neg. Fraction (%)
No Signature	+835 to +1781	+17267 to +40262	≈ 15.0 to 32.0	≈ 8.8 to 20.7
Type 1	+1008 to +1393	+23833 to +27537	≈ 18.1 to 25.1	≈ 12.3 to 14.2
Type 2	+1483 to +1726	+29890 to +44866	≈ 26.7 to 31.1	≈ 15.4 to 23.1

Table 4.1: Amount of converted positive intances and their corresponding label.

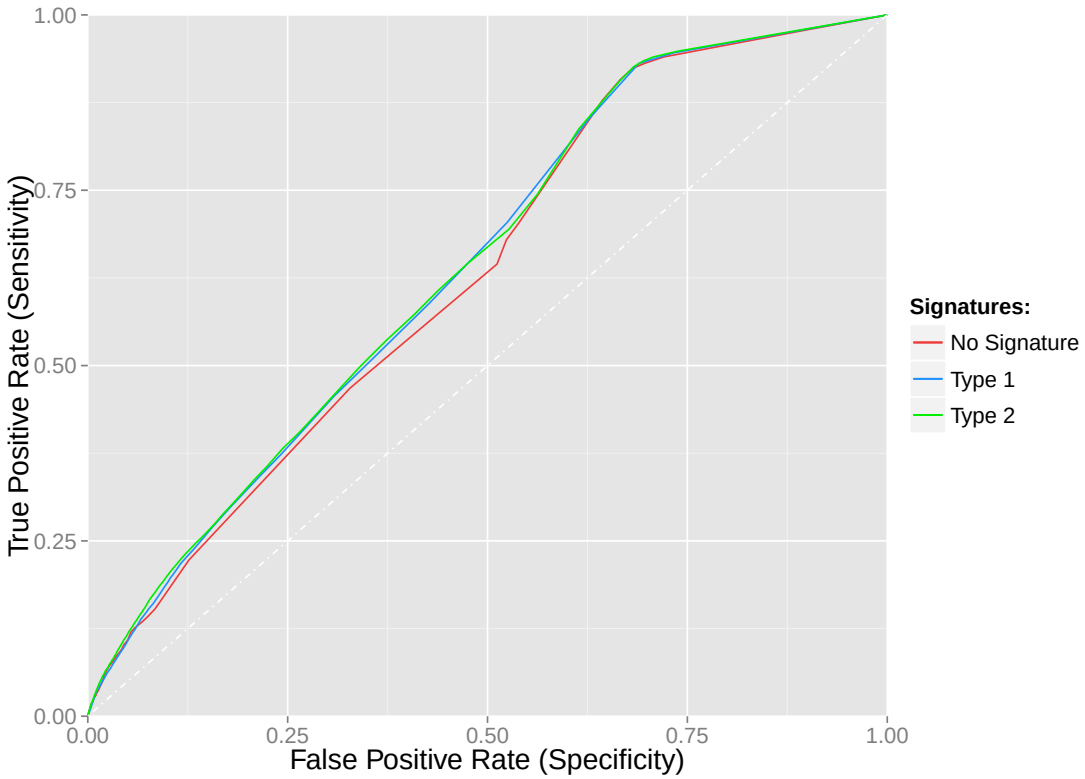

Figure 4.28: Normal Naive Bayes Model ROC Curve

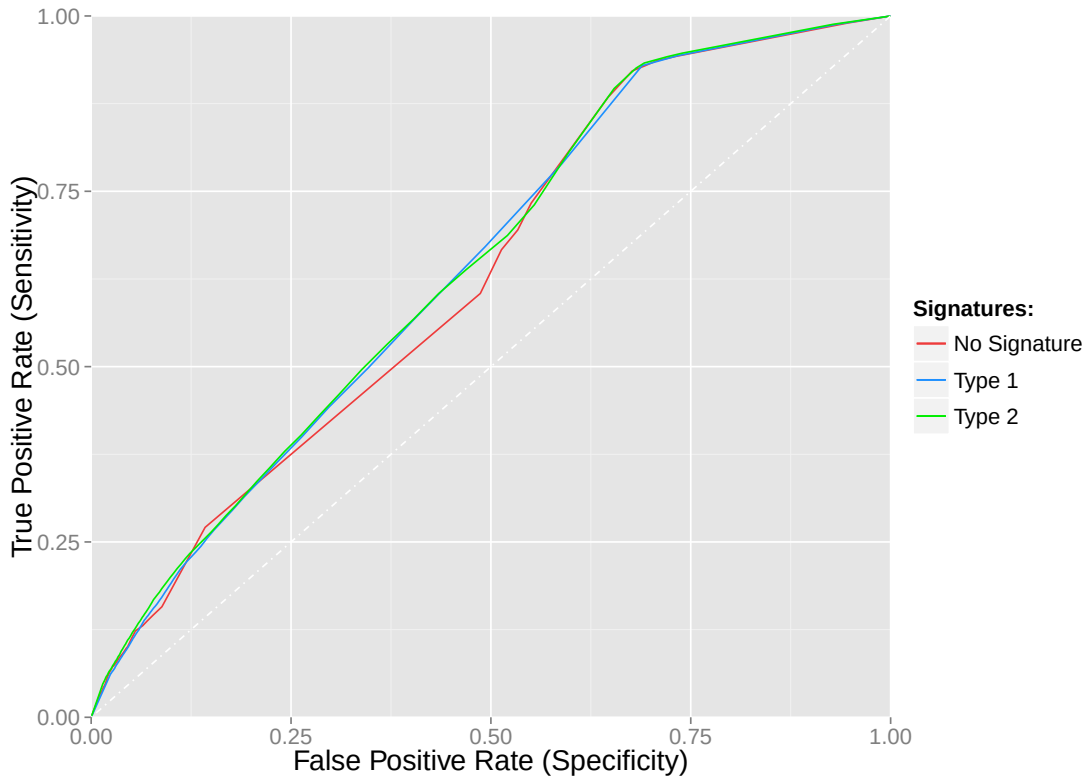

Figure 4.29: Maximal Positive Margin Naive Bayes Model ROC Curve

Chapter 5

Conclusions

This project presented an evaluation over the effect that unbalanced class datasets have in the classification algorithm Naive Bayes and measures to improve its performance. The analyses in the standard Naive Bayes algorithm revealed the expected poor classification performance. As a measure to improve the performance, one common method is to under sample the major class data. This method allowed to control number of examples in the major class by ajusting the *undersampling rate*. Controlling the class distribution translates in the control of the performance of the classifier. Better results in the *recall* were achieved in the lower *undersampling rate*, whit a trade-off with the *precision*.

Undersampling data before constructing the Naive Bayes classifier produces a trade-off between a good class distribution and a bad prior attribute distribution. The *Naive Class Distribution* suggests a different approach by allowing to calibrate the class distribution without compromising the prior attribute distribution. his solution showed some similar results to the undersample method.

The second proposal, *Naive Bayes Margin*, has the main objective of allowing more positive classification. The ideal behind its development is to evaluate the negative margin to the decision boundary and extend the positive margin without compromising the negative classification. This concept is similar to a change in the decision boundary, by moving the decison boundary in the negative direction, allowing for the positive margin to increase. This proposal results showed an improve in the *recall* with a trade-off with the *precision*.

Besides the algorithm proposed and tested, in the data preparation stage, some additional features were developed. These features improved in general all the models classification performance.

In summary, as expected, our results showed that the standard Naive Bayes is not appropriate for problems with an unbalanced class distribution. In the case study investigated in this project, the combination of this algorithm with undersampling improved its performance in terms of *recall* but not so much in terms of *precision*. The *Naive Class Distribution* and the *Naive Bayes Margin* algorithms proposed in this project yielded results that are comparable with the combination of standard Naive Bayes with undersampling. This shows that the proposed algorithms successfully

address the issues of unbalanced class distributions. Concerning the case study investigated, the results confirm that problem-specific data transformation can deeply affect the performance of the classification models.

5.1 Future Work

A deeper investigation should be made concerning the behaviour and performance of the proposed solutions. The *Naive Class Distribution* proposal can be improved in the method used to control the *Prior Percentage Distribution*. In this work, the method used to evaluate the performance is by trying different values and evaluating the results. An automated process that could calibrate this value in a faster way to obtain a better classification.

The *Naive Bayes Margin* proposal, can be improved by creating further conditions in the application of the *marging values*. The process of application of these values to the resulting predictions, only uses information regarding the classes. Some additional components, like other attributes information, can be used to refine the margin values to be applied to the prediction.

Other possibility is to test the *Naive Bayes Margin* method in other types of classifiers since the only requisite is for the classifier to output class probabilities for each instance. In the end, the viability of this solutions should be tested in other types of data.

References

[1] PA Estévez, CM Held, and CA Perez. Subscription fraud prevention in telecommunications using fuzzy rules and neural networks. *Expert Systems with Applications*, 2006.

[2] E Rosas and Cesar Analide. Telecommunications Fraud: Problem Analysis-an Agent-based KDD Perspective. *epia2009.web.ua.pt*, 2009.

[3] Richard a. Becker, Chris Volinsky, and Allan R. Wilks. Fraud Detection in Telecommunications: History and Lessons Learned. *Technometrics*, 52(1):20–33, February 2010.

[4] Jaakko Hollmén and V Tresp. Call-Based Fraud Detection in Mobile Communication Networks Using a Hierarchical Regime-Switching Model. *Advances in Neural Information Processing Systems*, 1999.

[5] Constantinos Hilas and John Sahalos. An application of decision trees for rule extraction towards telecommunications fraud detection. 2007.

[6] Yufeng Kou and CT Lu. Survey of fraud detection techniques. *Networking, sensing, and control, 2004 IEEE international conference on Vol.2 IEEE*, pages 749–754, 2004.

[7] C Breen and CA Dahlbom. Signaling systems for control of telephone switching. *Bell System Technical Journal*, (September), 1960.

[8] T Fawcett and F Provost. Adaptive fraud detection. *Data mining and knowledge discovery*, 316:291–316, 1997.

[9] V Chandola, A Banerjee, and V Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 2009.

[10] Animesh Patcha and Jung-Min Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, August 2007.

[11] Clifton Phua, Vincent Lee, Kate Smith, and Ross Gayler. A comprehensive survey of data mining-based fraud detection research. *arXiv preprint arXiv:1009.6119*, 2010.

[12] J Han, M Kamber, and J Pei. *Data mining: concepts and techniques*. 2006.

[13] Constantinos S. Hilas. Designing an expert system for fraud detection in private telecommunications networks. *Expert Systems with Applications*, 36(9):11559–11569, 2009.

[14] Luis Torgo. An Infra-Structure for Performance Estimation and Experimental Comparison of Predictive Models. Technical report, 2013. URL: https://github.com/ltorgo/performanceEstimation.

[15] Foster Provost. Machine learning from imbalanced data sets 101. *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, 2000.

[16] D Mladenic and M Grobelnik. Feature selection for unbalanced class distribution and naive bayes. *ICML*, 1999.

[17] Hamid Farvaresh and Mohammad Mehdi Sepehri. A data mining framework for detecting subscription fraud in telecommunication. *Engineering Applications of Artificial Intelligence*, 24(1):182–194, 2011.

[18] Simon Augustin, Carmen Gaiß er, Julian Knauer, Michael Massoth, Katrin Piejko, David Rihm, and Torsten Wiens. Telephony Fraud Detection in Next Generation Networks. In *AICT 2012, The Eighth Advanced International Conference on Telecommunications*, pages 203–207, May 2012.

[19] Tom Fawcett and FJ Provost. Combining Data Mining and Machine Learning for Effective User Profiling. *KDD*, 1996.

[20] Volker Tresp Michiaki Taniguchi, Michael Haft, Jaakko Hollmén. Fraud detection in communications networks using neural and probabilistic methods. In *In Proceedings of IEEE International Con- ference in Acoustics, Speech and Signal Processing Vol. 2. IEEE Computer Society*, volume II, pages 1241–1244. 1998.

[21] KC Cox, SG Eick, GJ Wills, and RJ Brachman. Brief application description; visual data mining: Recognizing telephone calling fraud. *Data Mining and Knowledge Discovery 1.2*, pages 1–6, 1997.

[22] Scikit-Learn. Naive bayes, 2013. URL: http://scikit-learn.org/stable/modules/naive_bayes.html.

[23] Wikibooks. Data mining algorithms in r/classification/naïve bayes, May 2014. URL: http://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Classification/Na%C3%AFve_Bayes.

[24] Eibe Frank and RR Bouckaert. Naive bayes for text classification with unbalanced classes. *Knowledge Discovery in Databases: PKDD 2006*, 2006.

[25] GH John and Pat Langley. Estimating continuous distributions in Bayesian classifiers. *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence.*, 1995.