A Churn Prediction Model Based on Gaussian Processes

by

Mehdi Ben Lazreg

supervised by

Ole-Christofer Granmo

Master Thesis in
Information and Communication Technology

The University of Agder, Faculty of Engineering and Science
Department of Information and Communication Technology

Grimstad, June 2, 2013

Abstract

The telecommunication's market has changed to a saturated one during the past few years. This makes it relatively costly to acquire new customers than to retain existing ones. Recent research has further showed that identifying potential customers who intend to leave a service provider for another (churners) and offering them an incentive to keep their subscriptions can produce significant savings for the telecommunication company. However, in most cases, it turned out that most of the used techniques to solve this problem fails to address the complex relationship between customer features and churn. Therefore, they struggle to achieve acceptable prediction accuracy. This might be due to the fact that real customers' data are large, unbalanced, with missing values, and with non-linear dependencies between features. In this thesis, I investigated a new class of techniques for predicting customer churn based on Gaussian processes - a recent approach to regression and classification based on Bayesian reasoning principles. I developed a Gaussian process based model for churn prediction and tested it on a real customers' data belonging to a French telecommunication's company (*Orange*). The method obtained an area under the curve (AUC) score of 0.77. This score outperformed most of the previously used techniques that I evaluated on the same data set. The best of them had an AUC score of 0.73. To conclude, my empirical results proved that Gaussian processes can provide state-of-the-art performance in churn prediction, and the approach and results that I report offer a solid basis for further development of Gaussian processes for churn prediction.

Preface

This report is written in the context of IKT 590 Master thesis to fulfil a third semester requirement of master in information and communication technologies at the faculty of engineering and science at the University of Agder. The work on this project started on January 2013 under the supervision of professor Ole-Christofer Granmo. Since then I started studying the basic of Gaussian processes and how can we apply them to churn prediction. We were able to test our approach on a data set containing informations about the behaviour of a French telecommunication company's costumers and I succeeded to obtained encouraging results.
At the culmination of this work, I would like to take the opportunity to thank all those who helped me achieve the objectives of this thesis. I would also like to pay my gratitude to my project supervisor professor Ole-Christofer Granmo. His knowledge and experience of the field were remarkable. His guidance throughout the hole project, his precious technical advices and his directives and suggestions were crucial to the success of my work. Finally, I want to thank all the fellow students as well as the university's personnel and teachers who helped me in multiple ways during all the course of my studies at the university of Agder.

Grimstad
May, 2012
Mehdi Ben Lazreg

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Customers' churn prediction is the term adopted to describe the process of identifying customers who intend to quit a company and move to another service provider [8]. It has become, during the recent years, a relevant subject of data mining and has been investigated in different industries including banking [7] [19], life insurance [14] and telecommunication [22] [11] [10]....

Churning customers can be divided into two main groups: financial and commercial churners. In one hand, financial churn occurs when either the company or the customer unwillingly decides to cancel the subscription due to facts such as the customer can no longer afford to pay for the service. On the other hand, commercial churn takes place if the customers willingly decide to withdraw his subscription. This can be due to different reasons including that he or she found a cheaper offer in a concurrent company or realised that this latest disposes of a new technology that the original company does not have.... Although it is difficult to distinguish between the two categories just by looking at the company's customers' database. This is because they are mostly related to a privet aspect of the customer life, Burez and den Poel [3] made an attempt to differentiate between them in the case of a pay-TV company. They concluded that financial churn are easier to detect.

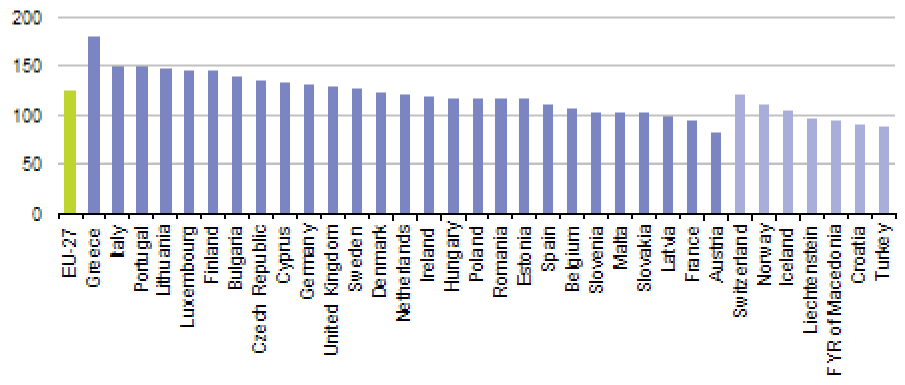


Figure 1.1: Percentage of mobile phone subscriptions per 100 inhabitant in different European countries in 2009

Source: Eurostat

1.1 Motivation

The telecommunication's markets changed, in the last decade, from a fast growing market to a saturated one. For example, the average percentage of mobile phone subscription per 100 inhabitant is around 125% in the European union countries. It is around 110% in Norway (figure 1.1) which means that there are more phone subscriptions than inhabitant. This makes it more difficult to acquire new customers and pouches many telecommunication's companies to try to focus more and more on retaining the existing ones. Moreover, Mozer et al. [15] showed that using predictive techniques to identifying potential churner and offer them an incentive to keep their subscriptions can produce significant savings for a telecommunication company.

Another motivation to our work is the fact that Gaussian processes are a rich and powerful learning tool that allows to predict the likelihood of customers to churn. It has not been used previously to predict churn. This thesis will, then, be a small step in the direction of adapting Gaussian process to churn prediction.

1.2 Thesis' objectives

In most cases, real customers' data sets are large, unbalanced, with missing values, and non-linear dependencies between features. Furthermore, a combination of numerical and categorical features is typical. This makes churn prediction a challenging data mining problem. Therefore, the first goal of this thesis is to work on the data-set by using some pre-processing techniques to deal with missing value as well as features' selection. This treatment will allow a better performance for any machine learning algorithm [8].

Hadden et al. [8] also showed that state-of-the-art learning algorithm struggles to produce acceptable performance when applied to churn prediction. On the other hand, Gaussian processes learning is a rich technique that tries to predict the function that relates the features and the target value. It can handle more complex relationship between features and target function. Besides, Rasmussen and Williams [21] showed that under certain condition, techniques including logistic regression, neural network as well as support vector machines can considered as special case of Gaussian process.

Therefore, This thesis proposes to improve and adapt Gaussian processes to churn prediction. It will try to answer the question whether this method can outperform most of the methods used in the field of churn prediction including logistic regression, decision tree, neural networks.... Moreover, Gaussian process presents a fundamental parameter called covariance function that defines, in our case, when the algorithm considers two customers to have a similar behaviour. This thesis will investigate whether using different covariance function for different type of features can improve the result of the classification.

1.3 Contributions

The thesis will try to resolve the issue related to the data set. These issues include missing value, constant features and highly correlated features. It will propose a features selection technique based on the information gained by each feature about the target value. Besides, it will check a number of feature and their relation with the target value to discuss the difficulty of prediction churn based on these features.

Finally, it proposes Gaussian process as a novel approach for churn prediction. Gaussian processes are a reach and flexible method that deserves the investigation for this purpose. As it will be shown in chapter 2, Gaussian process were not used previously for churn prediction so I hope that this thesis will be a first and an encouraging step in that direction.

1.4 Thesis' agenda

This report is arranged as follows: chapter 2 will present an overview of the previous research done in the field of churn prediction.

Chapter 3 will introduce Gaussian process and how they can be used for regression and classification learning. Further chapter 4 will illustrate a solution to implement Gaussian process for churn prediction.

Chapter 5 presents a detailed description of steps followed to pre-process the data set and deal with its issues. A testing of the proposed approach on the preprocessed data will be performed in chapter 6. Farther, it will compare it with the other techniques developed for this purpose. The chapter will end by discussion whether the objectives describes in section 1.2 were answered.

The report will finally end by a conclusion summarising the main result of this thesis and proposing future work directions.

Chapter 2

Review of previous works

Many research have been conducted over the years in order to predict customers' churn. At first state of the art learning techniques has been investigated. Datta et al. [4] developed a churn analysis, modelling and prediction (CHAMP) model based on neural network. They used simple regression, nearest neighbour and decision trees as well. Nonetheless, their research could not distinguish a best method.

Hung et al. [10] also used decision tree and neural network to predict customers' churn. They tried to evaluate the performance of each method by applying them to detect the churners among the customers of a telecommunication's company. Those methods were applied every month on a data set that was also monthly updated over a one-year period. Their result showed a height performance fluctuation due to the variation of the customers' behaviour during that period and they could not establish a relationship between customer behaviour and the target value (churn in this case). Moreover, it should be acknowledged that the nature of a real-customers' data set makes the problem really challenging for the state of the art machine learning techniques used in this case.

To address this problem, more advanced learning techniques started to be explored. Hwang et al. [11] discovered that logistic regression performed better than neural network and decision tree for churn prediction. However, it should be mentioned that they generated clusters of customers based on their past and future spendings. Then, they used those clusters as an input of the classification.

To deal with the non-linear dependency between feature, Yi and Guo-en [23] developed a support vector machine (SVM) model. SVM allows to model complex relationships between feature. It produces as output the likelihood of customers churn. The model outperformed logistic regression, neural network, decision tree and Bayesian network when applied to predict churn for a telecommunication's company.

Au et al. [2] argued that beside knowing whether a customer is going to churn or not, it is important to discover the likelihood of the customers doing so. This allows the companies to target customer with a height churn probability by the mean of retention campaign. For this purpose, they developed a technique based on a combination of rules learning and genetic algorithms. The method outperformed neural networks and decision tree.

Verbeke et al. [22] have developed a profit centric performance measure to evaluate customer churn prediction techniques. This method is based on the loss that a company can undergo if a misclassification occurs. For example, if a churner has been evaluated as non-churner, the company will not take action to retain him, he will then leave the company and the company will lose all the money that the customer would have spent if he stayed. Those losses are accumulated and used as a measure for the classification techniques. Alternating decision tree (ADT a method based on combining different derision tree classification algorithms) and neural network has been distinguished from the other used classification techniques based on this criterion.

Furthermore, Niculescu-Mizil et al. [17] used the same data set that I have been able to get in the context of the knowledge discovery in data competition [5]. They used in the purpose of discovering churners a method based on ensemble selection. They started from a set of multiple classifiers, then, tried to found out which subset of those classifiers provides the better result. During this process they also discovered that logistic regression provides the best individual result and combined with other classifiers can perform even better.

Mozer et al. [15], in the other hand, showed that a non-linear neural network outperformed logistic regression in prediction churn. They concluded that a sophisticated non-linear neural network can better exploit non-linear structure in the data than a linear based logistic regression method. They also suggested as future research topic that using Gaussian process would help improve the prediction's accuracy.

Finally, Gaussian processes has been used broadly for regression and classification problems that contain complex relationship between features and the target function [21]. It produced good result in robot dynamics learning [16], speech recognition [18] and human action recognition [24]. It has not been used for churn prediction. Nevertheless, SVM, neural network and logistic regression that proved to be good prediction techniques for churn as shown previously [15] [11] [23] are considered by Rasmussen and Williams [21] as special cases of Gaussian process under certain conditions. Rasmussen and Williams [21] further explains the relatively narrow use of Gaussian process by the fact that it requires handling large matrix and its inversion, which is a complex problem that need to be addressed especially for large data sets.

Chapter 3

A review of Gaussian process learning

Supervised learning is a machine learning task that tries to infer a function from a labelled training data. Thus, given a set of observation $D = \{(x_i, y_i) \,|i = 1...n\}$ where y_i is a label associated to x_i, supervised learning attempt to discover the function f such that $f(x_i) = y_i$ for all observations x_i. In order to perform this task, some assumption must be made about the function f otherwise any function that is consistent with the training data could be accepted. One way to do this would be to restrict the class of considered function by taking, for example, into account only linear functions. This approach could be limited especially if the labels and the observations are not linearly related. Another way would be to give a prior probability to each function where higher probabilities are assigned to functions that are considered more likely. This method, nonetheless, seems to have a serious problem since we are dealing with an infinite number of functions each one of them needs to be assigned a prior probability. This is where Gaussian process intervene.

This chapter will try to introduce Gaussian process (sections 3.1 and 3.2). Further, section 3.3 will present how it can be used for regression learning. Finally, section 3.4 will demonstrate how Gaussian process can be approximated and used for classification learning.

3.1 Gaussian process

Gaussian process is a generalisation of the Gaussian probability distribution. While a probability distribution describes a random variable which can be a scalar or a vector, a Gaussian process governs the properties of functions [21]. A set of functions is defined by simply defining a Gaussian process.

Definition 1 *For any set S, a Gaussian process on S is a set of random variables $(f(t), t \in S)$ such that $\forall n \in \mathbb{N}, \forall t_1 ... t_n \in S\ (f(t_1), ..., f(t_n))$ is a multi-variant Gaussian.*

A Gaussian process is fully defined by its mean function:

$$\mu(t) = \mathbb{E}[f(t)] \tag{3.1}$$

and its covariance or kernel function:

$$k(t, s) = \mathbb{E}[(f(t) - \mu(t))(f(s) - \mu(s))] \tag{3.2}$$

In the other hand:

Theorem 1 *Let S be a set,*
$\mu : S \to \mathbb{R}$ a mean function and
$k : S \times S \to \mathbb{R}$ a covariance function
Then, There exists a Gaussian process $(f(t))_{t \in S}$ such that:
$\forall s, t \in S; \mathbb{E}[f(t)] = \mu(t)$ and $\mathbb{E}[(f(t) - \mu(t))(f(s) - \mu(s))] = k(t, s)$

The previous theorem states that a Gaussian process is simply generated by well defining a mean and a covariance functions. Thus, those two functions specify a set of infinite function by the mean of that Gaussian process. Although the mean is important component of a Gaussian process it neither controls the shape nor the class of the function generated by the Gaussian process so it often taken as zero for simplicity reasons [21]. The covariance function on the other hand needs to be studied deeper.

3.2 Covariance function

The covariance function is the most important element of a Gaussian process. It encodes the assumption about the function that has to be learned[21]: It controls the shape and the nature of the function governed by the Gaussian process (continued, discontinued...). From another perspective, the covariance function defines when two elements are considered similar, which is a critical aspect of supervised learning: if some set of elements having the same target value y are considered similar to another element x then it is more likely that x will have the same target value.

Definition 2 *A covariance function is a function $k : S \times S \to \mathbb{R}$ such that:*
$\forall n \in \mathbb{N}; \forall x_1, ..., x_n \in S$ *the matrix* $C = (c_{i,j} = k(x_i, x_j))$ *is positive semi-definite*
$\Leftrightarrow \forall u \in \mathbb{R}^n; u^t C u \geq 0$

An infinity of functions can conform to the specification of the previous definition. However, the most significant and used function can be organized into two categories [21].

3.2.1 Stationary covariance function

A stationary covariance function is a function of the difference $r = x_1 - x_2$ of two value x_1 and x_2 of a feature. These functions have the characteristic of being invariant to any translation of the inputs. The most used stationary covariance function are:

The γ-exponential covariance function: it has the following equation:

$$k(r) = \exp\left(-(\frac{r}{l})^{\gamma}\right); 0 < \gamma \leq 2 \tag{3.3}$$

The parameter l defines a characteristic length-scale which controls what should be the minimum distance between x_1 and x_2 for them to be considered uncorrelated. For large value of l, the covariance function will be independent of r. Figure 3.1 shows an example of a function generated by the γ-exponential kernel for $\gamma = 0.5$. It can be seen that for this value of γ, the function generated by the Gaussian process is discontinued.

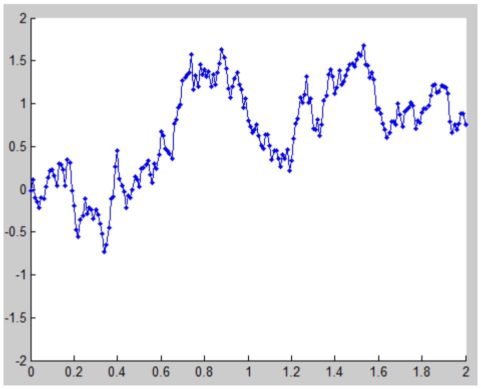


Figure 3.1: Example of a function generated by the The γ-exponential kernel for $\gamma = 0.5$

If the value of γ equals to 2, this covariance function is called the squared exponential covariance function. The kernel has the following expression:

$$k(r) = \exp\left(-\frac{r^2}{l^2}\right) \tag{3.4}$$

It is the most famous covariance function. This function is infinitely differentiable which means that the functions generated by a Gaussian process with this covariance function is very smooth. An illustration of this is presented in figure 3.2.

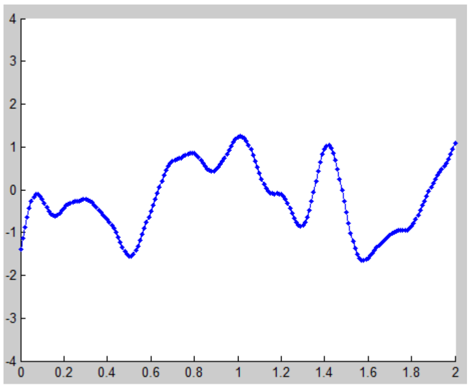


Figure 3.2: Example of function generated by the squared exponential kernel

Another notable special case of this kernel is the case where $\gamma = 1$, the covariance function is called the exponential covariance function. Figure 3.3 shows an example of a function generated by the exponential kernel. It can be seen that the function is discontinuous and had an exponential shape.

The Marten class of covariance functions: it has the following expression:

$$k(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu r}}{l}\right)^{\nu} K_{\nu}\left(\frac{\sqrt{2\nu r}}{l}\right) \tag{3.5}$$

Where K_ν is a modified Bessel function (see A.1 for more detail), Γ is the Euler gamma function and l is a free parameter of the covariance function. This process can model a less smoother or discontinues Gaussian processes. It should be noted

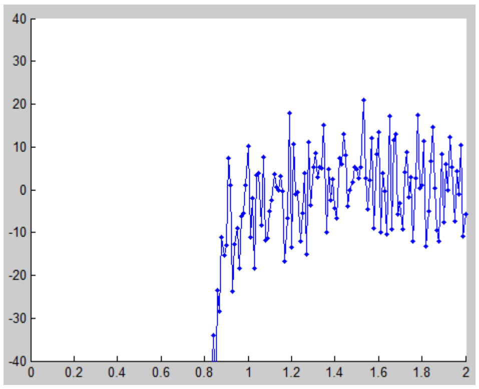


Figure 3.3: Example of function generated exponential kernel

the exponential covariance functions can also be obtained from this kernel when $\nu = 1/2$.

The rational quadratic covariance function: It has the following expression:

$$k(r) = \left(1 + \frac{r^2}{2\alpha l^2}\right)^{-\alpha} \tag{3.6}$$

Where α and l are free parameters. This function can be seen as the infinite sum of squared exponential covariance function with different length-scale l since:

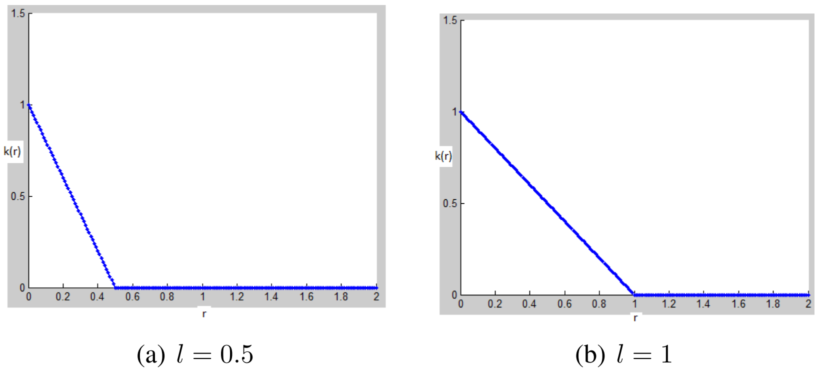
$$\left(1 + \frac{r^2}{2\alpha l^2}\right)^{-\alpha} = \int h(l) \exp\left(-r^2/2l^2\right) dl$$

Where h is a given function.

The piecewise polynomial covariance function: It has the following expression:

$$k(r) = max(0, 1 - \frac{\|r\|}{l}) \tag{3.7}$$

Where l is a free hyperparameter. The piecewise polynomial is a kernel with compact support which means that the covariance between two points will be equal to zero if they distance exceeds a threshold specified by l. Figure 3.4 illustrate the value taken by a piecewise polynomial kernel as a function of r for different l .



(a) $l = 0.5$ (b) $l = 1$

Figure 3.4: Value taken by a piecewise polynomial kernel in function of r

White noise covariance function: it has the following expression:

$$k(r) = \sigma\delta(r) \tag{3.8}$$

Where δ is the Dirac delta function. Figure 3.5 shows an example of a function generated by the white noise kernel for $\sigma = 1$. The figure shows a randomly generated noise function centred around 0.

3.2.2 The dot product covariance function

The dot product covariance functions are a functions of the product x_1x_2 of two value x_1 and x_2 of an input parameter. These function have the characteristic of being invariant to rotation of the coordinate around the origin. They are mostly used to model a polynomial relationship between the features and the target value.

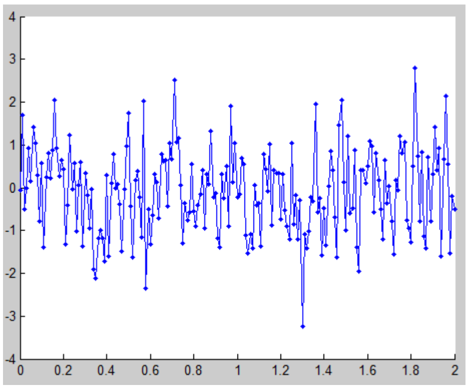
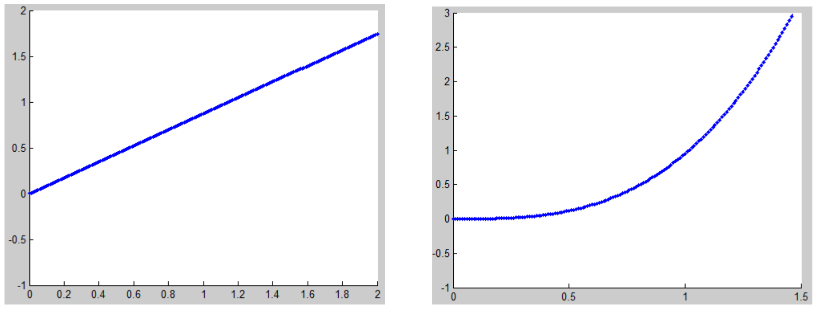


Figure 3.5: Example of a function generated by the The white noise kernel for $\sigma = 1$

The most used dot product covariance function is:

$$k(r) = \left(\sigma_0^2 + x_1 x_2\right)^p ; p \in \mathbb{N} \tag{3.9}$$

Figure 3.6 shows the set of function generated by the covariance function of the previous equation for different choices of p.



(a) $p = 1$ and $\sigma = 0$ (b) $p = 3$ and $\sigma = 0$

Figure 3.6: Example of a set of function generated by dot product kernel

Table 3.1 presents a summary of the most commonly used covariance function. It should be noticed that each covariance function has a number free parameters also known as hyperparameters. In the example of the squared exponential

Table 3.1: Summary of several commonly used covariance functions

Covariance function	**Expression**
Squared exponential	$\exp\left(-\frac{r^2}{2l^2}\right)$
Marten class	$\frac{2^{1-\nu}}{\Gamma(\nu)}\left(\frac{\sqrt{2\nu r}}{l}\right)^{\nu} K_\nu\left(\frac{\sqrt{2\nu r}}{l}\right)$
Exponential	$\exp\left(-\frac{r}{l}\right)$
γ-exponential	$\exp\left(-(\frac{r}{l})^{\gamma}\right)$
The rational quadratic	$\left(1+\frac{r^2}{2\alpha l^2}\right)^{-\alpha}$
piecewise polynomial	$max(0, 1-\frac{\|r\|}{l})$
White noise	$\sigma\delta(r)$
The dot product	$(\sigma_0^2 + x_1 x_2)^p$

covariance function the hyperparemeter is l. In the case of the dot product covariance function they are σ and p. Those parameters need to be carefully chosen and can affect the accuracy of the predictive model. Section 4.2 will describe how those parameter are chosen in the case of churn prediction. Finally, it should be noted that other covariance functions can be obtained by combining the previously cited functions following the subsequent propriety. The resulting function includes all the proprieties of the combined functions [21].

Propriety 1 *If k_1 and k_2 are two covariance functions then the following functions are also covariance function:*

- $\alpha k_1; \forall \alpha \geq 0$
- $k_1 + k_2$
- $k_1 * k_2$

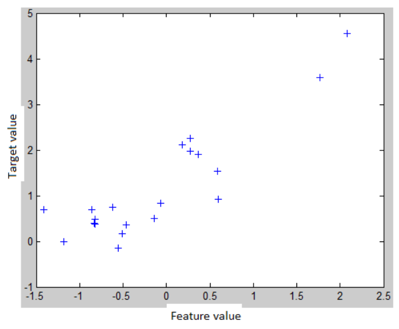
3.3 Gaussian Process regression

We have seen in the beginning of this chapter that Gaussian process learning tries to identify the function that relates the features and the target value. To intuitively present this process, let us consider the simple case of one feature having different value for different instance. Each instance also have a target value. Figure 3.7(a) illustrates this scenario. The points in the figure correspond to the target value as function of the feature's value for a given instances. Gaussian process regression tries to learn the most likely function that is consistent with those points. The shape and the nature (continued, discontinued, smooth...) of this function is controlled by the choice of the covariance function. This choice presents our prior knowledge about the function to be learned. It can be made by studying the data and discovering some patterns that could lead us to lean toward a certain kind of functions. Figures 3.7(b) and 3.7(c) illustrate the functions learned by two different Gaussian process with different covariance functions (squared exponential in figure 3.7(b) and Marten class in figure 3.7(c)). It can be remarked that the function learned using the Marten class kernel is less smooth than the one learned using the squared exponential kernel. This due to the nature of the two kernel explained in section 3.2.1. Finally, when the target value t of a new instance needs to be learned, we just need to apply the learned function to the value of its feature x like presented in figures 3.7(d).

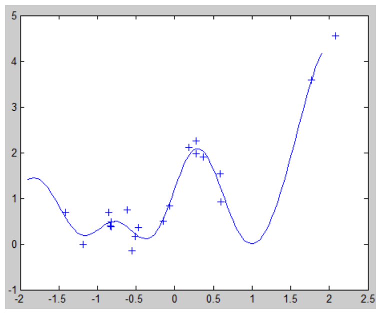
To put this into a more general and mathematical way, let us consider $D = \{(x_1, y_1)...(x_n, y_n)\}$ to be a data set where $(x_i)_{1\leq i\leq n} \in \mathbb{R}^d$ is the set of values taken by a number of features for a specific instance and $(y_i)_{1\leq i\leq n} \in \mathbb{R}$ the target value. In Gaussian process regression, we suppose that [21]:

$$y_i = f(x_i); 1 \leq i \leq n \tag{3.10}$$

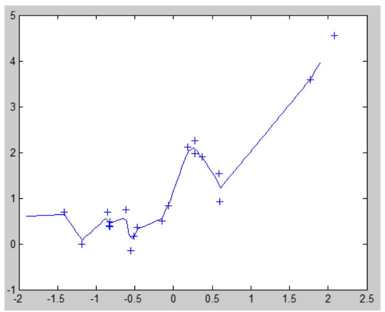
Where $\left(f(x_i), x_i \in \mathbb{R}^d\right)$ is a Gaussian process on $\mathbb{R}^d$. Further, let us assume that this Gaussian process has a mean function μ and a covariance function k



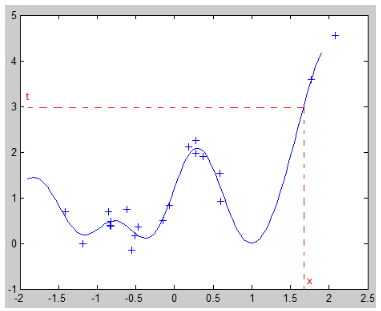
(a) The target's values as a function of the respective feature's values



(b) Learned function using the squared exponential kernel



(c) Learned function using the Marten class kernel



(d) Learning the target value for a new instance

Figure 3.7: Example of a Gaussian process regression

and take an integer number p such that $0 < p < n$. Moreover, Suppose that $y_b = (y_{p+1}, ..., y_n)$ are the observed target value and that we want to infer $y_a = (y_1, ..., y_p)$.
In Gaussian process inference the problem is reduced to determine the probability $p(y_a|y_b)$.
Or equation 3.10 states that $y_i = f(x_i)$ and $\left(f(x_i), x_i \in \mathbb{R}^d\right)$ is a Gaussian process on $\mathbb{R}^d$ so definition 1 gives that $(f(x_1), ..., f(x_n))$ is a multi-variant Gaussian. This implies that $Y = (y_1, ..., y_n)$ is also a multi-variant Gaussian $(Y \sim N(\tilde{\mu}, K))$ where:

$$\tilde{\mu} = (\mu(x_1), ..., \mu(x_n))^t \tag{3.11}$$

$$K = \begin{pmatrix} k(x_1, x_1) & k(x_1, x_2) & \cdots & k(x_1, x_n) \\ k(x_2, x_1) & k(x_2, x_2) & \cdots & k(x_,2x_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_n, x_1) & k(x_n, x_2) & \cdots & k(x_n, x_n) \end{pmatrix} \tag{3.12}$$

Let suppose that:

- $\tilde{\mu} = (\mu_a, \mu_b)^t$ where:
  - $\mu_a = (\mu(x_1), ..., \mu(x_p))^t$.
  - $\mu_b = (\mu(x_{p+1}), ..., \mu(x_n))^t$.
- $K = \begin{pmatrix} K_{a,a} & K_{a,b} \\ K_{b,a} & K_{b,b} \end{pmatrix}$ where:
  - $K_{a,a} = \begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_p) \\ \vdots & \ddots & \vdots \\ k(x_p, x_1) & \cdots & k(x_p, x_p) \end{pmatrix}$.
  - $K_{a,b} = \begin{pmatrix} k(x_1, x_{p+1}) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_p, x_{p+1}) & \cdots & k(x_p, x_n) \end{pmatrix}$.
  - $K_{b,a} = \begin{pmatrix} k(x_{p+1}, x_1) & \cdots & k(x_{p+1}, x_p) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_p) \end{pmatrix}$.
  - $K_{b,b} = \begin{pmatrix} k(x_{p+1}, x_{p+1}) & \cdots & k(x_{p+1}, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_{p+1}) & \cdots & k(x_n, x_n) \end{pmatrix}$.

Since $Y = (y_a, y_b)$ is a multivariate Gaussian, then $(y_a|y_b)$ is a multi-variant Gaussian [21] $(y_a|y_b) \sim N(m, C)$ where:

$$m = \mu_a + K_{a,b}K_{b,b}^{-1}(y_b - \mu_b) \tag{3.13}$$

$$C = K_{a,a} - K_{a,b}K_{b,b}^{-1}K_{b,a} \tag{3.14}$$

Up to this point, we showed that the predictive distribution of the target value y_a corresponding to a novel test inputs is a Gaussian with a mean and variance given respectively by equations 3.13 and 3.14. Once we compute $K_{b,b}^{-1}$, calculating the mean and the variance is straightforward. However, when we are dealing with a big training set (over 10000 training sample [21]) computing $K_{b,b}^{-1}$ becomes a resource and time-consuming problem. The purpose of this section was to present an application of Gaussian process to regression problem that we can use as an analogy to the classification problem.

3.4 Gaussian Process classification

In this section, we will discuss binary Gaussian process classification where target y_i can take the value 1 or -1. The aim of the classification problem is to determine the probability $p(y_a|y_b)$. In the regression problem, we supposed that $y_i = f(x_i)$ where $\big(f(x_i), x_i \in \mathbb{R}^d\big)$ is a Gaussian process on $\mathbb{R}^d$. Thus, $f(x_i)$ can take any value in $\mathbb{R}$. The basic idea of Gaussian classification is to turn the output of the regression model to a class probability by using a response function that squashes the output into the interval $[0, 1]$ [21]. This grantees a valid probabilistic interpretation. An example of response function can be the logistic function:

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{3.15}$$

Therefore, for Gaussian process classification we suppose that [21]:

$$y_i = \sigma(f(x_i)) \tag{3.16}$$

In Gaussian process regression, the set $(f(x_1), ..., f(x_n))$ is a multi-variant Gaussian which allow us to get the result of equation 3.13 and 3.14. However, in the classification problem the set $(\sigma(f(x_1)), ..., \sigma(f(x_n)))$ has a non-Gaussian likelihood, which makes the problem of determining $p(y_a|y_b)$ intractable [21]. Hence, approximation method are used to approximate this distribution to a Gaussian distribution. This makes the computation of this likelihood less complex. The most used approximation methods are Laplace approximation and expectation propagation.

3.4.1 The Laplace approximation

The Laplace approximation is based on the second order Taylor expansion of $\ln p(y_a|y_b)$ around its maximum[21]. This maximum exists indeed (prove of its existence can be found in A.2). Lets supposed that it is obtained for y_0 then, the expansion would give:

$$\begin{aligned}\ln(p(y_a|y_b)) &= \ln p(y_0|y_b) + \frac{1}{2}\nabla\nabla\ln(p(y_0|y_b))(y_a - y_0)^2 \\ \Rightarrow p(y_a|y_b) &= \exp(\ln p(y_0|y_b))\exp(\frac{1}{2}\nabla\nabla\ln(p(y_0|y_b))(y_a - y_0)^2)\end{aligned}$$

Let $F(y_0) = \ln p(y_0|y_b)$ and $\frac{1}{\sigma^2} = -\nabla\nabla\ln p(y_0|y_b)$ then

$$p(y_a|y_b) = \exp(F(y_0))\exp(\frac{1}{2}\frac{(y_a - y_0)^2}{\sigma^2}) \tag{3.17}$$

The probability distribution $p(y_a|y_b)$ has an expression analogue to a Gaussian distribution with a mean y_0 and variance σ^2. Thus, $y_a|y_b$ is locally approximated to a Gaussian distribution $\sim N(y_0, \sigma^2)$.

3.4.2 The expectation propagation

The expectation propagation approximation starts from the fact that:

$$p(y_a|y_b) = \frac{p(y_b|y_a)p(y_a)}{p(y_b)} = \frac{1}{p(y_b)}p(y_a)\prod_{j=p+1}^{n} p(y_j|y_a) \tag{3.18}$$

Let g be a function such that:

$$\begin{cases} g_p(y_a) = p(y_a) \\ g_j(y_a) = p(y_j|y_a) \end{cases}$$

Equation 3.18 gives:

$$p(y_a|y_b) = \frac{1}{p(y_b)}\prod_{j=p}^{n} g_j(y_a) \tag{3.19}$$

Where $p(y_b) = \int p(y_a, y_b)dy_a = \int \prod_{j=p+1}^{n} g_j(y_a)dy_a$.

By consequence, The hole problem comes down to determining the function $g_j(y_a)$ for all j.

The expectation propagation approximates $g_j(y_a)$ with a member of the exponential family of functions $\widetilde{g}_j(y_a)$ [21](More detail about the exponential family of functions can be found in A.3). The result is a tractable function $q(y_a)$ (equation 3.20) that minimise an error metric between $q(y_a)$ and $p(y_a|y_b)$. The error metric chosen in most cases is the Kullback-Leibler divergence (see A.4 for more detail)

$$q(y_a) = \frac{1}{\int \prod_{j=p+1}^{n} \widetilde{g}_j(y_a) dy_a} \prod_{j=p+1}^{n} \widetilde{g}_j(y_a) \tag{3.20}$$

It should be noted that the expectation propagation approximation is a more sophisticated approach and has been proven to provide better results in practice that the Laplace approximation [21] since it chooses to proximate the posterior probability $p(y_a|y_b)$ with a function that minimize an error metric. Whereas, the Laplace approximation locally approximate the posterior using a second order Taylor expansion. However, the expectation propagation is more complex and requires expensive computations compared to the Laplace method.

Chapter 4

Churn prediction using Gaussian processes

After studying the basic aspect of Gaussian processes, this chapter will describe how can we use them on our data set to predict churn. In section 4.1, I will try to present an overview of the proposed solution. Chapter 3 proposed different types of covariance functions that can be used in Gaussian process learning. Each covariance function contains different parameters (referred to also as hyperparamters) that need to be carefully chosen in order to build a accurate prediction model. Therefore, section 4.2 of this chapter will to explain the choice of the covariance functions and section 4.3 the choice of the hyperparamters.

4.1 Proposed solution

The data are composed of two types of features: numerical and categorical. Furthermore, we showed in section 3.2 that covariance function defines when two customers are considered having the same behaviour based on the values taken by the features for those customers. To further explain this point, let us suppose that one of the numerical features is the number of call made by a customer and one of the categorical features contains the subscription type (prepaid, billed subscription,...). This latest is mapped to a number as we will see is section 5.3 with each number presenting a type of subscription. Lets now consider two customers with different number of calls made and different subscriptions' type and that we are using a covariance function consisting on the difference between the value taken by each feature to discover how close the behaviour of those customers is. In the case of the numerical feature, the difference between the number of calls made can be a good criteria to distinguish between the two customers' behaviours. However, In the case of the categorical feature, the difference between two randomly picked numbers representing the type of subscription would not hold a significant meaning and can, by consequence, harm the prediction process. But using a covariance function that takes the value zero if the customers have different subscription's types and one otherwise can tell us if the two customers have the same subscription which could be a meaningful information.

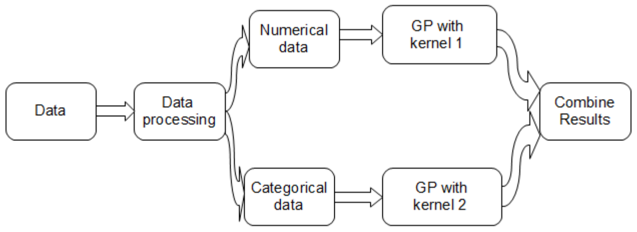


Figure 4.1: Solution's steps

I propose, then, as solution to the churn prediction problem to use different covariance functions for each type of features (numerical and categorical). Figure 4.1 shows the proposed solution. After the data processing, I spilt the data into two sets one containing the categorical feature and the other the numerical features. Then, I apply two Gaussian processes with different covariance functions to each type of feature. Since the result produced by each process is a churning probability, combining the results will consist on averaging the probabilities obtained from each one.

A final choice needs to be made concerning whether to use the Laplace approximation or the expectation propagation approximation in for the Gaussian process classification. Since the expectation propagation is a more sophisticated approach and has been proven to provide better results in practice that the Laplace approximation in many cases [21], I decided to use it for churn prediction.

4.2 Model selection

As seen in the previous section, the choice of the covariance function is an important issue. Therefore, a special care need to be taken to perform this choice. In the case of numerical features a stationary covariance function (section 3.2.1) can be the best choice. Nonetheless, the study of the data set that will be performed in chapter 5 did not provide any result that could lead to lean toward any of the covariance function. Consequently, in the experiments that will be done later, I will try different covariance functions and empirically discover the best one.

In the case of categorical features, as we discussed previously, the difference between the value of a feature would not have any significance. Thus, a function like the piecewise polynomial covariance function (section 3.2.1) that take the value 0 if the difference between the value of the feature exceeds a certain threshold could be a suitable choice if this latest is set to 1. In this case, if two customers have different feature's values this function would return the value 0.

4.3 Adaptation of hyperparameters

We sow in section 3.2 that for a certain features the covariance function does not only depend on the values taken by that feature but also on a number of free parameters. For example, in the case of a squared exponential covariance function, we consider a feature taking the values x_1 and x_2 among others. The covariance function can be expressed as follow:

$$k(x_1, x_2) = \exp\left(-\frac{(x_1 - x_2)^2}{l^2}\right) \tag{4.1}$$

Where the length-scale l represents the free parameter in this case. However, in our case we are dealing with a multitude of features and we want to apply the same covariance function -let's say the squared exponential- to all of them. To solve this problem let us suppose the number of features to which we want to apply the same covariance function is m and x_s and x_t two vectors in $\mathbb{R}^{\gtrdot}$. The elements of x_s and x_t represent the value taken respectively by each feature for two customer s and t. In this case, the squared exponential covariance function can be expressed as follows:

$$k(x_s, x_t) = \exp\left(-(x_s - x_t)^t M(x_s - x_t)\right) \tag{4.2}$$

Where $M = diag(l_1^{-2}, l_2^{-2}, ..., l_m^{-2})$ is a diagonal matrix containing the value of different length-scale for each feature. The problem now is to find the value of M that best fits the data and the prediction we need to make. One approach could be to take different values of the elements of M corresponding to each feature. In this scenario, we are giving less importance to features assigned larger length-scale. Since, as seen in section 3.2.1, for large length-scale the covariance function will independent from that feature. This can be considered a kind of feature selection process. The other approach would be to take the same value for all the elements of M. In this case, all the features will be considered having the same importance. However, for each case the hyperparameters need to be learned from the training set. To do so, let us consider in a more general case that $\Theta = (\Theta_1, \Theta_2, ..., \Theta_e)$ is a vector containing all the hyperparamters of a covariance function k and we follow the notations of chapter 3 where y_b are the observed target. In the classification case, $p(y_b)$ is approximated to a multivariate Gaussian $N(\mu_b, K_{b,b})$ (section 3.4).

where:

$$K_{b,b} \approx \begin{pmatrix} k(x_{p+1}, x_{p+1}) & \cdots & k(x_{p+1}, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_{p+1}) & \cdots & k(x_n, x_n) \end{pmatrix}; \mu_b = (\mu(x_{p+1}), \mu(x_{p+2}), ..., \mu(x_n))$$

Thus:

$$p(y_b) = \frac{1}{\sqrt{(2\pi)^b \left|K_{b,b}\right|}} \exp\left(-\frac{1}{2}(y_b - \mu_b)^t K_{b,b}^{-1}(y_b - \mu_b)\right) \tag{4.3}$$

$$\Rightarrow \ln p(y_b) = -\frac{n-p}{2}\ln(2\pi) - \frac{1}{2}\ln(|K_b|) - \frac{1}{2}(y_b - \mu_b)^t K_b^{-1}(y_b - \mu_b) \tag{4.4}$$

Since $K_{b,b}$ is a function of Θ, the problem comes down the finding the value of Θ that maximise $\ln p(y_b)$. That implies finding Θ_0 for which:

$$\frac{\delta}{\delta\Theta}\ln p(y_b) = -\frac{1}{2}tr(K_b^{-1}\frac{\delta K_b}{\delta\Theta}) + \frac{1}{2}(y_b - \mu_b)^t K_b^{-1}\frac{\delta K_b}{\delta\Theta}K_b^{-1}(y_b - \mu_b) = 0 \tag{4.5}$$

This approach allows to choose the hyperparamater that best fit the training data. However, it is likely that we are in a case with multiple local optima each one of them representing a different interpretation of the data and we end up choosing a value of Θ corresponding to one of local optima. Nonetheless, Rasmussen and Williams [21] states that choosing a value of Θ in a local optima does not highly affects the prediction's accuracy.

Chapter 5

Data preprocessing

In a case where the data set contains irrelevant, redundant or noisy data, the knowledge discovery process during the training phase becomes difficult. In this chapter, I will describe the first step of the solution presented in the previous chapter which data preprocessing. I will start by presenting an overview of the data set used for the classification (section 5.1). Section 5.3 demonstrate how the irrelevant features are filtered from the data. Finally, section 5.4 presents an analysis of the selected features. All the data procession done in this chapter were performed using *KNIME* a workbench for data mining and processing [1].

5.1 Description of the data set

The data set used in this thesis was provided by the French telecommunication company *Orange* in the context of the 2009 edition of the knowledge discovery in data competition (KDD cup 2009) [5]. The data contain 50000 instance and 230 features. It also includes a binary target variable corresponding to churn. The target takes the value 1 if the person is a churner and -1 otherwise. In addition to that the data set has the following characteristic:

- A large number of missing values (about 60%).
- Unbalanced class proportions: the number of churners is much less then the number of non-churners (7.3% of the costumer in the data set are churners).
- Noisy data.
- A combination of numerical and categorical features (190 numerical features and 40 categorical features).

For confidentiality reasons and to protect the privet information of its customers, *Orange* has made the following modification on the data set:

- The features names were discarded and changed (the name of the features are henceforth: feature1, feature2,...,feature230).
- Multiplying each numerical feature by a random factor.
- Categorical features changed and mapped with randomly generated category names.

5.2 Data preprocessing scheme

Due to the proprieties of the data set illustrated in the previous section, some processing need to be done before carrying through any feature selection method. Figure 5.1 illustrates the different steps of the data processing. The first task was to replace the missing value. In the numerical features the missing numerical value were replaced with the average value of the feature in question. In the categorical feature, on the other hand, the missing value were replaced by the string *'missing'*. Moreover, since Gaussian processes does not support learning for categorical features, I transformed all the strings in the categorical features to number using a bijective mapping between the set in a way that a string is represented by a unique number in the resulting data set.

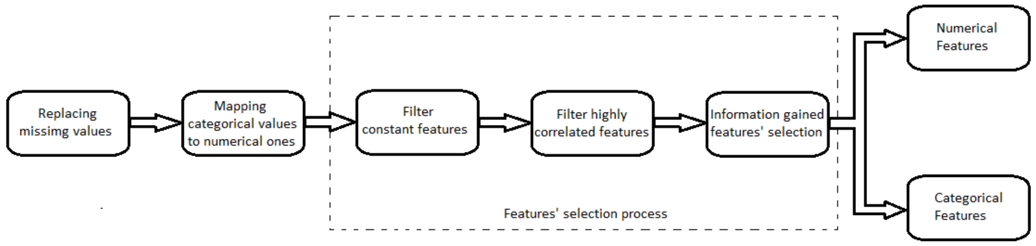

Replacing missimg values
Mapping categorical values to numerical ones
Filter constant features
Filter highly correlated features
Information gained features' selection
Features' selection process
Numerical Features
Categorical Features

Figure 5.1: Date processing scheme

5.3 Feature selection

Features' selection consists on selecting from the data set the best set of features to use for the learning process. 'Different combinations of data hold different analytical powers'[8]. This means that depending on the prediction that need to be performed, a set of features that better fit this prediction have to be chosen. By consequence, it is necessary to identify the features that best suits the type of prediction to be carried through. In our case, the best features needed to perform costumers' churn prediction have to be filtered. A special attention should be paid to feature selection because it helps reducing the amount of data to be used by keeping the important features and deleting the redundant and less informative ones. Moreover, the feature's quality will influence the accuracy of the predictive model.

Then, I noticed that the data contain constant feature. So, I applied a filter that excludes the constant features from the data set. This operation reduced the number of features in the data set from 230 to 176.

Further, I applied a correlation filter to data set. This operation allows to filter the redundant features. It consists on computing, for each pair of feature, a correlation coefficient that measures the correlation between two features. The computed coefficient is called the Pearson product-moment correlation coefficient (PCC). It is based on dividing the covariance between two feature by the product of their stander deviation. This coefficient is a number between -1 and 1, the closer this coefficient is to one the more correlated the features are. The features that have a PPC above a specific threshold are filtered. I set the threshold to 0.9 and this allowed to cut the number of features to 147.

Finally, in order to discover the worth of each feature and select the best ones to perform the classification, I applied a filter method to measure the information gained by each feature with respect to the target variable (In this case churn) [13]. It, then, orders the features from the more to the least important one by assign to each feature a weight representing the information gained.

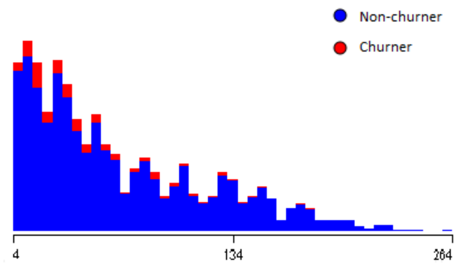
The information gained is the change of information entropy between the target variable and the target variable knowing the value of a specific feature is given. Let y be the target variable and x a features. The information gained is given by:

$$IG(y, x) = H(y) - H(y|x) \tag{5.1}$$

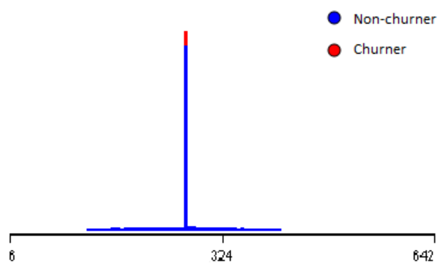
Where $H(y) = \mathbb{E}(-\ln p(y))$ is the entropy of y. This operation allows to discover that 88 of the previously selected 147 features bring near zero information about the target and are, then, eliminated. We are left with 59 features (41 numerical features and 18 categorical).

5.4 Data analysis

In this section, I will try to analyse some of the selected feature and their relation with each other and with the target value. First, I started with a look at the most informative of the features selected in the previous section. Figure 5.2 presents the number of occurrence of each value of feature51 and feature144 among the customers and the proportion of churning customer (in red) for each value. It should be noted by the look at the figure that those features -as well as the majority of the features in the data set- follow a Gaussian distribution with a small variance in feature144 and a larger one for feature52. We can also figure out why feature144 was one of the most informative feature: all the churner have the value 270 in this feature.



(a) Feature51



(b) Feature144

Figure 5.2: Visual of the two of the most informative feature and their relation with the target value

Then, I tried to make a Self-organizing map (SOM) presentation of the data. It is a data exploring technique base on neural network. This technique is used to present on a two-dimensional space (called map) a highly dimensional instances of a data. SOM is composed of set of neurons, each neuron has an assigned weight, which is initialised to a small or random value. The input data is then fed to the neuron and the weight of each neuron is adjusted until it feats the input data [12]. At the end of the process each neuron will present a cluster of customers that are considered to have similar behaviour. This process will allow to verify if the selected features present any clear separation between the churners and the non-churners.

In the map (figure 5.3), The neuron are portrayed by pentagons and the circles inside each pentagon exhibit the cluster of customers represented by the neuron and the proportion of churner in each one. The darker the pentagon is the larger the distance between the customers represented by it and the ones represented by its neighbours. As the figure shows, a huge majority of the pentagons are white which means that, by looking at the features, no clear distinction exist between the cluster of customers represented by these pentagon and no clear border exist between the two categories of customers (churners and non-churners).

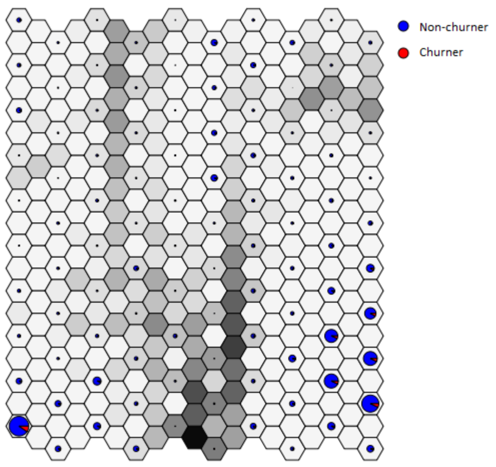


Figure 5.3: SOM presentation of the data

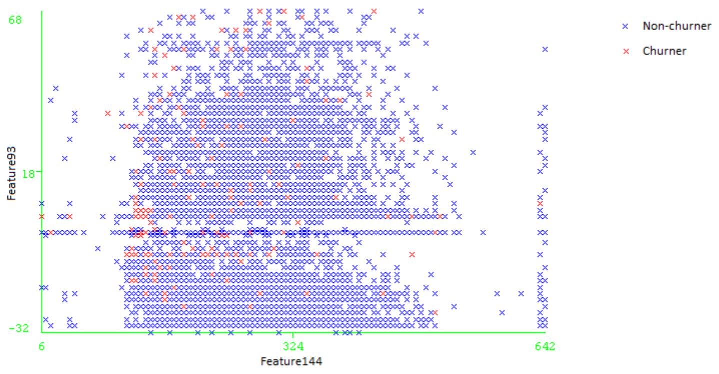


Figure 5.4: 2D Presentation of features 144 and 93

Further, I tried to study the information that a combination of two feature can bring about the target value. For that purpose, I made a 2D plot representing a certain feature as a function of another. The results for different combination of feature is illustrated in figures 5.5, 5.6 and 5.4 where the red cross represent the churning customers and the blue cross the non-churning ones. As shown by the figures, no clear separation was discovered the two category of customers. In deed, the churning customers are scattered over all the possible values of the features.

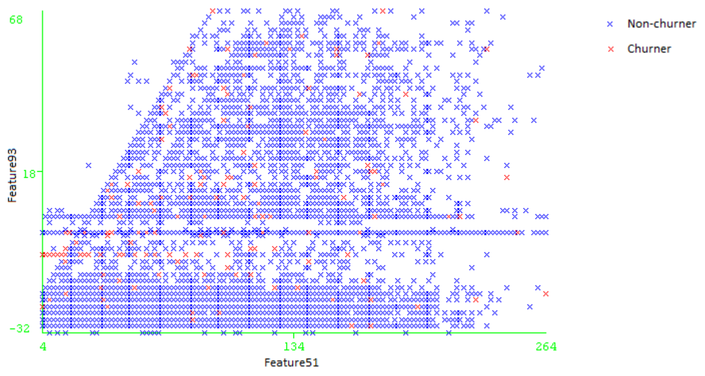


Figure 5.5: 2D Presentation of features 51 and 93

Finally, the result of this section show that the correlation between the features and the target value is not important. This makes the problem of training for the data set and prediction the churning customers rather difficult.

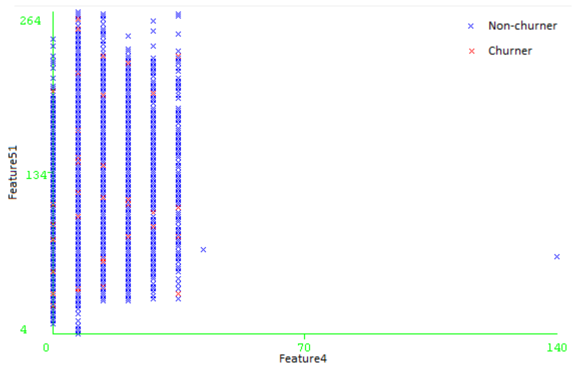


Figure 5.6: 2D Presentation of features 51 and 4

Chapter 6

Experiments and evaluation

In this chapter, I will try to apply the Gaussian process classification on the data set after preprocessing. I used a Matlab implementation of Gaussian process developed by Rasmussen and Nickisch [20]. During the experiments, I will test different choices of covariance functions and compared the results of the respective predictions. In order to do that we need a better accuracy measurement criteria than the percentage of correct classification because -as we have seen in chapter 5- the data contain 7.3% churners. Thus, a classification algorithm that classifies all customers as non-churner will end up with 92.7% accuracy according to the percentage of correct classification criteria. That is a height score that does not reflect how poorly the algorithm performed. Section 6.1 will try to answer to this problem. Section 6.2 will expose and discuss the result obtained by Gaussian process classification. Finally, section 6.3 will try to compare those results to those of the previously used methods for churn prediction.

6.1 Accuracy measurement criteria

The percentage of correct classification metric does not constitute, as seen previously, a proper accuracy measurement. The problem with that method arise when the churners are a minority in the data set. What we actually need is a criterion that would capture how many churners were detected by the classification algorithm.

The receiver operating characteristic (ROC) curve, on the other hand, is a graphical plot that allow to represent the true positive rate also known as sensitivity (in our case the number of churning customers detected by the classification algorithm divided by the number of true churners) versus the false positive rate also known as specificity (in our case the number of churning customers not detected by the classification algorithm divide by the true number of non-churners). If the predictions are the churning probability p and we suppose that the customer is classified as churner if $p > S$ for $0 < S < 1$ then the ROC curve present the sensitivity as a function of the specificity for each value of S.

In this case, the area under the ROC curve (AUC) could be a good performance measure in our case since it allows to reduce the ROC curve performance to a single number representing the expected performance of the classification. A random method will have an identity line ROC cure with an AUC of 0.5. Classification algorithm are supposed to have AUC higher than this. The bigger the AUC is the higher the classification' performance is.

6.2 Experiments and discussion

The data set contains 50000 customers. For the experiments, I divided it into two sets:

- Training set containing the third of data: this set is used to train the classification algorithm and to evaluate the mean and the hyperparameters of the

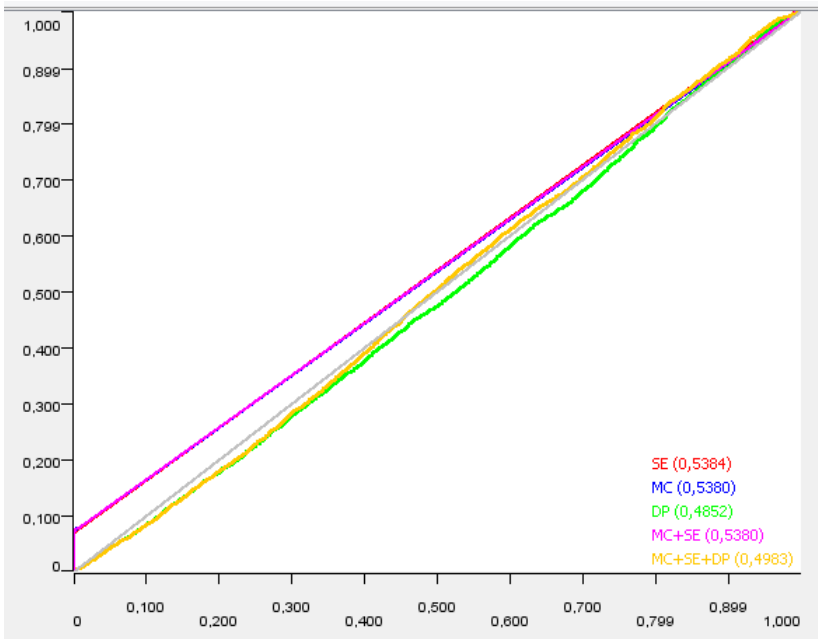


Figure 6.1: Results of Gaussian process classification applied to all the data set

covariance function.

- Testing set containing the rest of the data: this set is used to test the model build with the training set.

After the data preprocessing of chapter 5, we obtained a data set containing 59 features. In our first experiment we took the nine most informative ones containing a combination of numerical and categorical features (5 numerical and 4 categorical features) and tested the straightforward approach of applying the Gaussian process with different covariance functions. I tried the squared exponential (SE equation 3.4), Marten class (MC equation 3.5) and dot product (DP equation 3.9) covariance functions as well as the sum of MC and SE covariance functions and the sum of MC, SE and DP covariance functions. The result presented in figure 6.1. The performance is rather poor with the best method having an AUC of 0.5384 with the SE kernel. We can also notice that summing the different co-

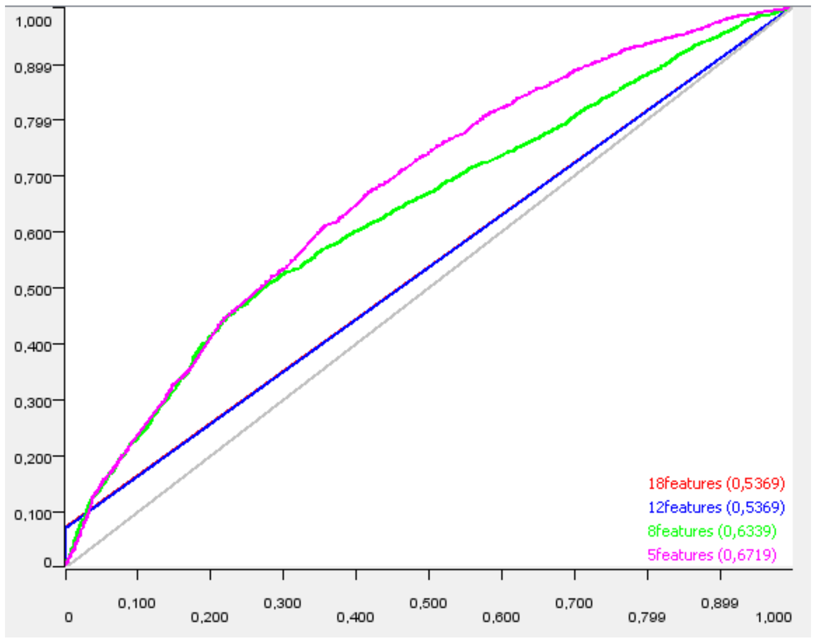


Figure 6.2: Results of Gaussian process classification applied with different numbers of features

variance function did not have the effect of improving the results given by the SE kernel.

The next step was -as suggested in section 4.1- to separate the categorical and numerical features. However, I wanted to discover the optimal number of feature that I can use from the selected 59 of each type (categorical and numerical). To do that, I used the MC covariance function on a different number of numerical features. The results are presented in figure 6.2 where the best result was obtained with the five most informative numerical features with an AUC of 0.67. The same thing was done with the categorical feature using the piecewise polynomial covariance function (PP equation 3.7) and the best result was obtained with the four most informative features.

Further, I used on the five numerical and four categorical features separately different covariance functions including the SE, MC, DP, and PP covariance function. The performances are illustrated in figure 6.3 and 6.4. The result shows

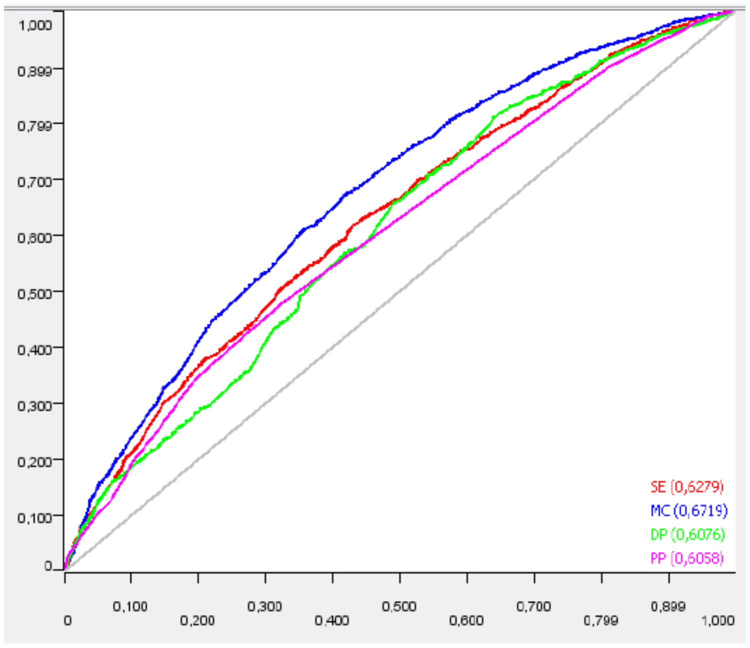


Figure 6.3: Results of Gaussian process classification applied to the numerical features

that the MC covariance function perform better with the numerical feature with an AUC of 0.67 and the PP covariance function provides a better accuracy with the categorical features with an AUC of 0.56. Each of those result is better than the result obtained in the first experiment (figure 6.1) where I used a Gaussian process classification on a combination of the same features. The best performance obtained so far is by Gaussian process applied on the numerical features alone. However, we can see that it struggles to make prediction out of the categorical features alone. This might be caused either by the possibility that Gaussian process leaning model is not suited to learning from categorical features or that the features themselves do not carry meaningful information about the target value.

The next step was to average the results giver by the Gaussian processes with the MC covariance function applied to numerical features alone and the PP covariance function processes applied to categorical features alone. The results are illustrated in figure 6.5. The combination of the two Gaussian processes

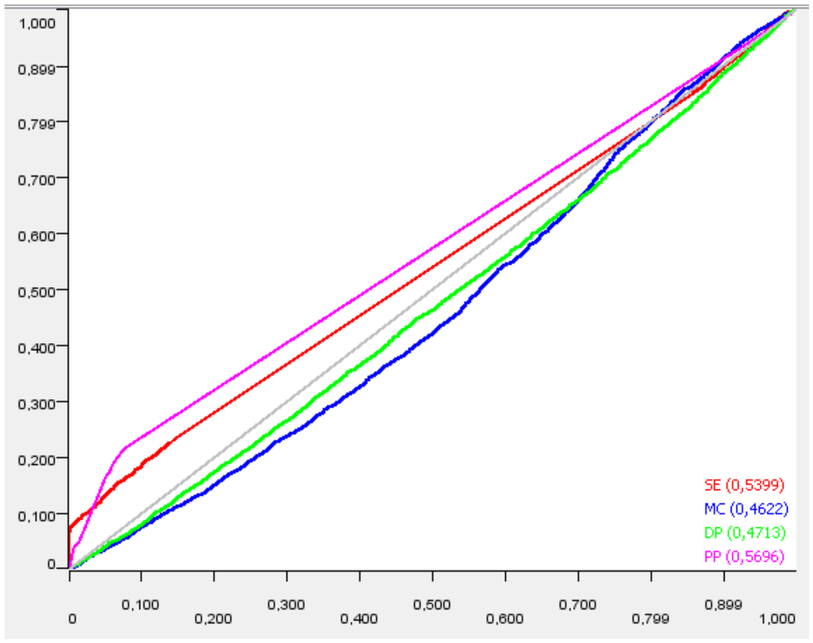


Figure 6.4: Results of Gaussian process classification applied to the categorical features

(AUC=0.68) did not outperform by much the result obtained from the Gaussian process applied to numerical features alone. Nevertheless, it allowed to avoid the weak performance obtained by using a unique Gaussian process on all the features (figure 6.1) and surpass it by much. This result responds to the one of the research questions of this thesis namely: is using different Gaussian processes with distinct Kernels each on a different type of data could be beneficial for churn prediction? However, we still want to improve those results.

We have seen in chapter 5 that the data contain some noise. Therefore, in our next experience we tried to add a white covariance (WN equation 3.8) function to the MC and PP covariance functions used previously. This addition turned out to be highly beneficial as shows figure 6.6. The AUC was 0.77 and it was the best performance obtained during our experiment. Nonetheless, it still needs to be compared with other prediction's methods to evaluate its worth.

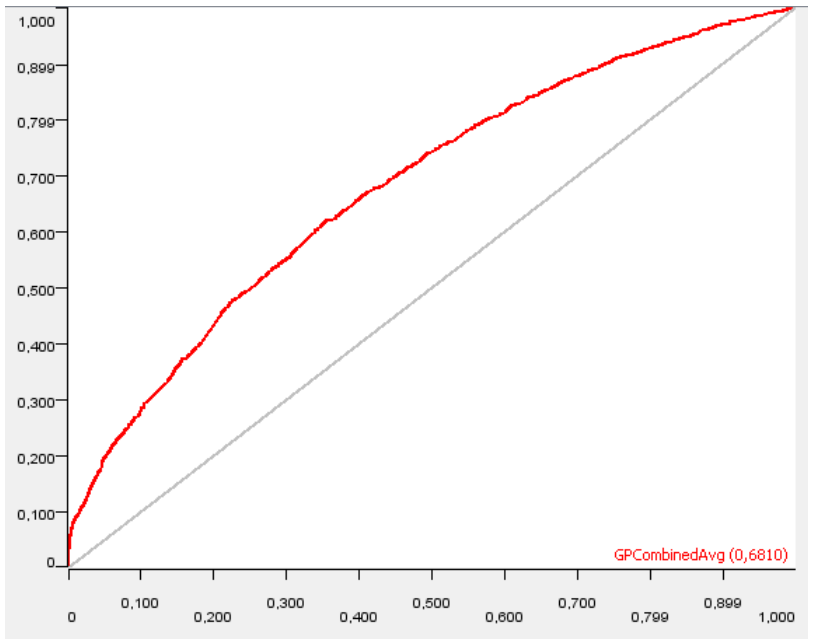


Figure 6.5: Result of the combination of two Gaussian processes

6.3 Evaluation of the results

In order to evaluate our result we tried to compare them with the results that most of the previously used methods for churn prediction obtained on our data set. Those methods are:

Decision tree learning: it is based on creating a decision tree (figure 6.7). Each node in the tree represent a feature. Based on the value of this feature for a specific customer a decision to choose the next node to move to is made. This process is repeated until the leaf node representing the class prediction is reached [13].

Alternating decision tree: it is a method based on combining different derision tree classification algorithms. In the case of decision tree learning in each node a decision is made based on the value of one feature. In the case of

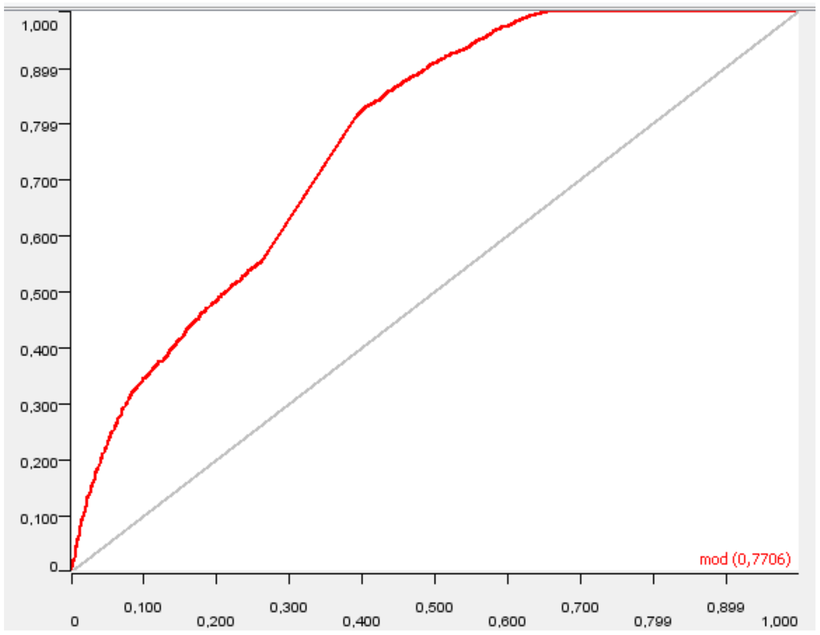


Figure 6.6: Result of the combination of two Gaussian processes with white noise

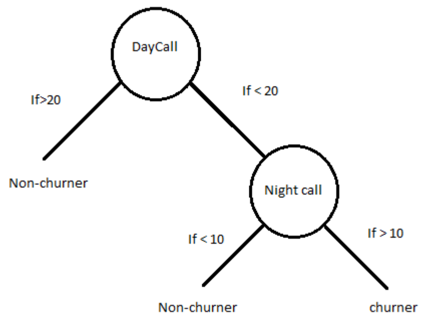


Figure 6.7: Example of decision tree learning tree for learning costumers' churn

alternating decision tree learning at each node the decision is made based on a vote between decision over multiple feature [6].

Artificial neural network: it is a set of entities called neurons that takes a set of input variables and tries to change it structure during the learning phase in order to find a relationship between the input variables and the target variable. Those relationships are used to predict the target variable [13].

Nave Bayesian classifier: it is based on the Bayesian theorem with the assumption that all the input variables or observation $\{O_1, O_2, ..., O_n\}$ are independent [13]. Lest assume that the target variable takes its value from the set $H = \{h_1, h_2, ..., h_m\}$. The nave Bayesian classifier tries to find the value of target variable that maximize the maximum a posteriori hypothesis:

$$Argmax_{h_j \in H}(P(O_1, O_2, ..., O_n|h_j)P(h_j)) \tag{6.1}$$

K nearest neighbour: it goes through a training data and, for each value of the target function, it label all the instances in the data set having this value with the same label. This procedure creates clusters of instances. It then goes ahead and measure the distance (in most cases Euclidean distance) between a new instance and all the cluster and find to which cluster it is the closest. The value of the target function that this cluster have is given to the new instance [13].

Logistic regression: it is based on the assumption the target value y_i is related to the values taken by the features x_i by the relation [21]:

$$y_i = \frac{1}{1 + \exp(-x_i)} \tag{6.2}$$

Support vector machine(SVM): it is a special case of Gaussian process where the kernel is [21]:

$$K(x_i, x_j) = (x_i x_j)^d; d \in \mathbb{N} \tag{6.3}$$

Ensemble selection: it is based on the idea that combining multiple predictive models could outperform any of those model used separately. In the experiments, I used a combination of random forests, boosted trees, logistic regression, SVM, decision trees, naive Bayes and KNN. This method was

Prediction method	**AUC**
Decision tree	0.54
Alternating decision tree	0.7
Artificial neural network	0.649
Nave Bayesian	0.644
K nearest neighbour	0.685
Logistic regression	0.696
Support vector machine	0.7
Ensemble selection	0.73

Table 6.1: Result of other prediction methods

the winning method of the KDD challenge from which we obtained our data set.

I used an implementation of those methods available in Knime[1] and Weka: a popular machine learning software developed in Java and containing a variety of learning algorithms[9]. The results of all those method are presented in table 6.1. The best of the alternative methods provides a performance of 0.73, but still not as good as the result obtained with our approach using Gaussian process. I should acknowledge that, for a lack of time, I did not go deep in the testing of those methods on the data set: I did not adapt their parameters to the problem in hand. Instead, I used a standard implementation of those methods. Moreover, I did not use a features selection process that select the best feature for each method. I rather used the most informative feature selected in chapter 5. This does not also mean that by using a feature selection process that adapt the selected features to each learning method we will end up with a totally different set of features. Thus, with more work, the result of the other tested method can be slightly improved and could be closer to the result obtained by the Gaussian process. This answer the question whether Gaussian process can outperform most of the method used in churn prediction, which is our second research question.

Chapter 7

Conclusion and further work

7.1 Conclusion

During this thesis, I studied Gaussian processes classification. Then, I investigate how can we use Gaussian process to predict customers churn. This allowed to develop a model for churn prediction based on separating the categorical and numerical features in the data set and use distinct Gaussian processes with different covariance functions on each.

The thesis started by processing the data to deal with missing value and irrelevant features. Then, I used a feature selection technique that allows to discover which feature brings the most information about churn. Finally, I used previously described Gaussian process classification model. The best results were obtained using a Marten class of covariance function and piecewise polynomial covariance function -in addition to a white noise covariance function- respectively on numerical and categorical features. The performance was an AUC score of 0.77. These results outperformed a Gaussian process with those same covariance functions, but used on both types of features as well as most of the tested churn prediction techniques.

I hope that this work will be an encouraging step toward applying Gaussian process for churn prediction in the telecommunication's industry and that it will motivate improving and enhancing this classification method and that it will broaden its use to may other areas.

7.2 Further work

The extension of this thesis can be done into two different directions. We can work on more data preprocessing that can help improve the results of the prediction. We can think of binning some rarely occurring value on certain features or work on combining exiting feature features to obtain new and more significant ones.

The second direction would be to work on the Gaussian process classification model itself. We have seen in chapter 6 that we did not obtain as good result with categorical as with numerical features and their combination did not improve by much the performance acquired by those latest. To improve this point, we can try to generate more sophisticated covariance functions that can capture better the relation between the features and the target value especially in the case of categorical features.

Appendix A

Mathematical background

A.1 Bessel function

A Bessel function is the solution $y(x)$ the Bessel differential equation:

$$x^2\frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x^2 - \nu^2)y = 0$$

The Bessel function of the first kind denoted as $J_\nu(x)$ are solution of the previous equation such that:

$$J_\nu(0) = c; c \in \mathbb{R} if \nu > 0 \lim_{x\to 0} J_\nu(x) = 0 if \nu < 0 \tag{A.1}$$

$J_\nu(x)$ can be expressed as follow:

$$J_\nu(x) = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!\Gamma(m+\nu+1)}(\frac{x}{2})^{2m+\nu}$$

Where $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$ is the Euler gamma function. The modified Bessel Bessel function has the following expression:

$$K_\nu(x) = \frac{\pi}{2}\frac{I_{-\nu}(x) - I_\nu(x)}{\sin(\nu\pi)}$$

Where $I_\nu(x) = i^{-\nu}J_\nu(ix)$ and i is the imaginary unit.

A.2 Concavity

A function f is concave if an only if its Hessian $\nabla\nabla f$ is a negative definite matrix. on the other hand:

$$p(y_a|y_b) = \frac{p(y_b|y_a)p(y_a)}{p(y_b)}$$

$p(y_b)$ does not depend of y_i so proving that $p(y_a|y_b)$ is concave comes down to proving that: $\nabla\nabla \ln p(y_b|y_a)p(y_i = 1)$ is negative definite. Or:

$$\ln p(y_b|y_a)p(y_a) = \ln p(y_b|y_a) + \ln p(y_a)$$

and we know that $p(y_a)$ is a multivariate Gaussian with:

$$p(y_a) = \frac{1}{\sqrt{(2\pi)^a\,|K_{a,a}|}}\exp\left(-\frac{1}{2}y_a^t K_{a,a}^{-1} y_a\right)$$

$$\Rightarrow \ln p(y_b|y_a)p(y_a) = \ln p(y_b|y_a) - \frac{a}{2}\ln(2\pi) - \frac{1}{2}\ln(|K_{a,a}|) - \frac{1}{2}y_a^t K_{a,a}^{-1} y_a$$

$$\nabla \ln p(y_b|y_a)p(y_a) = \nabla \ln p(y_b|y_a) - K_{a,a}^{-1}y_a$$

$$\nabla\nabla \ln p(y_b|y_a)p(y_a) = \nabla\nabla \ln p(y_b|y_a) - K_{a,a}^{-1} = -W - K_{a,a}^{-1}$$

Where $W = -\nabla\nabla \ln p(y_b|y_a)$. Note that $\nabla\nabla \ln p(y_b|y_a)$ is a diagonal matrix since y_i does not depend on y_j where $j \neq i$ and $y_i = \sigma(f(x_i))$ so as long as we choose a log concave function σ, W will be positive definite. Since $K_{a,a}^{-1}$ is positive semi-definite by definition of a covariance matrix, $\nabla\nabla \ln p(y_b|y_a)p(y_a)$ is a negative definite matrix and $p(y_a|y_b)$ is concave. This shows that the maximum exists.

A.3 The exponential family of functions

A distribution from the exponential family of functions of function has the form:

$$f_i(\theta) = h(\theta)g(\eta)\exp(\eta^t u(\theta))$$

Where η is a vector of hyperparameters. h, g and u are known functions. The normal distribution $\frac{1}{\sqrt{(2\pi\sigma^2)}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ is a special case of the exponential family of functions where:

$$\begin{aligned} h(\theta) &= \frac{1}{2\pi} \\ \eta &= (\frac{\mu}{\sigma^2}, -\frac{1}{2\sigma^2})^t \\ g(\eta) &= \frac{1}{|\sigma|}\exp(-\frac{\mu^2}{2\sigma^2}) \\ u(\theta) &= (\theta, \theta^2)^t \end{aligned}$$

A.4 Kullback-Leibler divergence

Let p and p be two probability density distribution such that:

- $p + q = 1$
- if $q(x) = 0$ then $p(x) = 0$

Then the Kullback-Leibler divergence between p and p is defined as:

$$D_{LK}(p\,||\,q) = \int_{\mathbb{R}} \ln(\frac{p(x)}{q(x)})p(x)dx$$

Appendix B

Glossary and Abbreviation

B.1 Glossary

∇: matrix differential

Γ: Euler gamma function

μ: mean function

Θ: vector of hyperparameters

D: data set

$\mathbb{E}$: expected value of a random variable

k: covariance function

K_ν: modified Bessel function

$N(m, C)$: multivariate Gaussian normal distribution with a mean m and covariance matrix C

p: probability distribution of a random variable

S: random space

t: transpose of a matrix

x_i: Set of observed features for a specific customer

y_i: The target value for a specific customer

B.2 Abbreviation

AUC: Area Under the Curve
CHAMP: CHurn Analysis, Modelling and Prediction
DP: Dot Product
GP: Gaussian Process
KDD: Knowledge and Discovery on Data
MC: Marten Class
PCC: Pearson product-moment Correlation Coefficient
PP: Piecewise Polynomial
ROC: Receiver Operating Characteristic
SE: Squared Exponential
SOM: Self-Organizing Map. SVM: Support Vector Machine
WN: White noise

Bibliography

[1] KNIME.com AG. Knime data mining workbench, February 2013. URL `http://www.knime.org/`.

[2] Wai-Ho Au, Keith C. C. Chan, and Xin Yao. A novel evolutionary data mining algorithm with applications to churn prediction. *IEEE transactions on evolutionary computation*, 7(6):1059–1066, December 2003.

[3] Jonathan Burez and Dirk Van den Poel. Separating financial from commercial customer churn: a modeling step towards resolving the conflict between the sales and credit department. *Expert Systems with Applications*, 35:497–514, 2008.

[4] Piew Datta, Brij Masand, Dr Mani, and Bin Li. Automated cellular modeling and prediction on a large scale. *Issues on the Application of Data Mining*, page 485502, 2001.

[5] Gideon Dror, Marc Boulle, Isabelle Guyon, Vincent Lemaire, and David Vogel. *Challenges in machine learning: the 2009 knowledge discovery in data competition (KDD cup 2009)*, volume 3 of *978-0-9719777-3-0*. Microtome Publishing, Brookline, Massachusetts, USA, 2011.

[6] Yoav Freund and Llew Mason. The alternating decision tree algorithm. *proceedings of the ieee*, 78, September 1990.

[7] Nicolas Glady, Bart Baesens, and Christophe Croux. Modeling churn using customer lifetime value. *European Journal of Operational Research*, 197: 402–411, 2009.

[8] John Hadden, Ashutosh Tiwari, Rajkumar Roy, and Dymitr Ruta. Computer assisted customer churn management: State-of-the-art and future trends. *Computers and Operations Research*, 34:2902–2917, 2005.

[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Exploration*, 11(1):1059–1066, 2009.

[10] Shin-Yuan Hung, David C. Yen, and Hsiu-Yu Wang. Applying data mining to telecom churn management. *Expert Systems with Applications*, 31:515–524, 2006.

[11] Hyunseok Hwang, Taesoo Jung, and Euiho Suh. An ltv model and customer segmentation based on customer value: a case study on the wireless telecommunication industry. *Expert Systems with Applications*, 26:181–188, 2004.

[12] Teuvo Kohonen. The self-organizing map. *proceedings of the ieee*, 78, September 1990.

[13] Tom Mitchell. *Machine learning*. 0-07-042807-7. McGraw-hill, international edition, 1997.

[14] K. Morik and H. Kopcke. Analysing customer churn in insurance data a case study. In *8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 325–336, New York, USA, 2004.

[15] Michael C. Mozer, Richard Wolniewicz, David B. Grimes, Eric Johnson, and Howard Kaushansky. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on Neural Networks*, 11:690–696, 2000.

[16] Duy Nguyen-Tuong and Jan Peters. Learning robot dynamics for computed torque control using local gaussian processes regression. In *ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems*, 2008.

[17] Alexandru Niculescu-Mizil, Claudia Perlich, Grzegorz Swirszcz, Vikas Sindhwani, Yan Liu, Prem Melville, Dong Wang, Jing Xiao, Jianying Hu, Moninder Singh, Wei Xiong Shang, and Yan Feng Zhu. Winning the kdd cup orange challenge with ensemble selection. In *JMLR Workshop and Conference Proceedings*, pages 23–34, 2009.

[18] Sunho Park and Seungjin Choi. Gaussian process regression for voice activity detection and speech enhancement. In *International Joint Conference on Neural Networks*, 2008.

[19] U. Devi Prasad and S. Madhavi. Prediction of churn behavior of bank customers using data mining tools. *Business Intelligence Journal*, 5:96–101, 2012.

[20] Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning toolbox, January 2013. URL `http://www.gaussianprocess.org/gpml/code/matlab/doc/index.html`.

[21] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. 026218253X. the MIT Press, 2006.

[22] Wouter Verbeke, Karel Dejaeger, David Martens, Joon Hur, and Bart Baesens. New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218:211–229, 2012.

[23] Li Yi and Xia Guo-en. The explanation of support vector machine in customer churn prediction. In *E-Product E-Service and E-Entertainment International Conference*, 2010.

[24] Hang Zhoua, Liang Wangb, and David Sutera. Human action recognition by feature-reduced gaussian process classification. *Pattern Recognition Letters*, 30(12):1059–1066, 1 September 2009.