ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΤΕΧΝΙΚΩΝ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΓΙΑ ΠΡΟΣΒΑΣΗ ΣΕ ΕΤΕΡΟΓΕΝΗ ΑΣΥΡΜΑΤΑ ΔΙΚΤΥΑ 5ΗΣ ΓΕΝΙΑΣ

Διπλωματική Εργασία

Πρασσάς Απόστολος

Επιβλέπων: Κοράκης Αθανάσιος

Βόλος 2019

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΤΕΧΝΙΚΩΝ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΓΙΑ ΠΡΟΣΒΑΣΗ ΣΕ ΕΤΕΡΟΓΕΝΗ ΑΣΥΡΜΑΤΑ ΔΙΚΤΥΑ 5ΗΣ ΓΕΝΙΑΣ

Διπλωματική Εργασία

Πρασσάς Απόστολος

Επιβλέπων: Κοράκης Αθάνασιος

Βόλος 2019

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DESIGN AND IMPLEMENTATION OF MACHINE LEARNING TECHNIQUES FOR 5G HETEROGENEOUS NETWORK ACCESS

Diploma Thesis

Prassas Apostolos

Supervisor: Korakis Athanasios

Volos 2019

Acknowledgements

First of all, I would like to thank my supervisor professor Korakis Athanasios for the knowledge he shared during my undergraduate studies and foremost the opportunity he gave me to work with him to complete this Thesis.

Also, I would like to thank PhD student Virgilios Passas and Postdoc Vasilis Miliotis for their excellent guidance, great support and kind advice throughout the accomplishment of this Thesis. It was an honor for me and a privilege to absorb a small chunk of their exceptional knowledge.

Super important was the support and the unconditional friendship and patience that my fellow students and friends provided me all these years. I am really grateful to all of them.

Last, but not least, I would like to express my deepest gratitude to my family who has always supported me with its own unique way to achieve my goals.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

«Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής».

Ο Δηλών

Πρασσάς Απόστολος

(Υπογραφή)

04/07/2019

Περίληψη

Η ραγδαία εξέλιξη της τεχνολογίας σε συνδυασμό με την τεράστια ποσότητα καθημερινών πληροφοριών, επιπρόσθετα της βελτίωσης και της εξέλιξης που έχει επιφέρει σε διάφορους τομείς της επιχειρηματικής δραστηριότητας, έχει ενισχύσει σημαντικά τον ανταγωνισμό μεταξύ των εταιριών. Ένα δημοφιλές πεδίο ανταγωνισμού αφορά τη συμπεριφορά των καταναλωτών να διακόπτουν ή όχι συνδρομές τους σε παρόχους ή υπηρεσίες, προκαλώντας σε αυτές αρνητικές συνέπειες. Σε αυτή τη διπλωματική εργασία, εστιάζουμε στη βιομηχανία τηλεπικοινωνιών και την τάση αποχώρησης των πελατών ή όχι από παρόχους τηλεπικοινωνιακών υπηρεσιών. Αναπτύσσοντας και δοκιμάζοντας έξι αλγόριθμους μηχανικής μάθησης, εφαρμόζουμε ένα μοντέλο που προβλέπει αν ένας πελάτης πρόκειται να αλλάξει ή όχι πάροχο. Επιπλέον, χειριζόμενοι πραγματικά δεδομένα, καλούμαστε να ξεπεράσουμε τις επιπτώσεις της ανισορροπίας των δεδομένων, καθώς και να επικεντρωθούμε στα χαρακτηριστικά των πελατών που τους επηρεάζουν περισσότερο στην απόφασή τους να αποχωρήσουν ή όχι από έναν πάροχο. Αξιολογώντας το προτεινόμενο μοντέλο και συγκρίνοντας, τα αποτελέσματα που προκύπτουν με αυτά ήδη δημοσιευμένων εργασίων, παρατηρούμε ότι επιτυγχάνουμε αντίστοιχη και σε κάποιες περιπτώσεις υψηλότερη ακρίβεια.

Abstract

The rapid development of technology combined with vast amount of daily information, in addition to the improvement and evolution it has brought to various areas of business, has greatly enhanced competition between companies. A popular field of competition concerns the behavior of consumers to discontinue their subscriptions to providers or services, provoking negative consequences. In this thesis, we focus on the Telecommunication industry and the customer churn, which refers to the customers' tendency to change telecommunications service providers. By developing and testing six machine learning algorithms, we implement a model that predicts customers churns. In addition, dealing with real-world data, we are called upon to overcome the effects of data imbalance, and to focus on the characteristics of customers that most influence them in deciding whether or not to leave a provider. By evaluating the proposed model and comparing the results with those of already published papers, we observe that we achieve corresponding and in some cases higher accuracy.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Description of Churn problem

The definition of churn is the event of customers leaving a company or a service. Also known as customer attrition, it is a crucial issue for service providers because it has negative effects on their revenue and reputation.

Although it is apparent that churn is a big deal in many industry fields, in this thesis, we focus on the churn problem in the Telecommunications industry. The rapid development of telecommunication technology results in many different services or 'tools' for users to deal with their daily communication. At the same time, the fact that customers have plenty of choices leads to an increase in competition between providers.

To shield against competition, a company should should be able to reveal the specific reasons for which customers decide to move on to another telecom company. There are two main categories of churners [1]. As depicted in Figure 1.1, these are voluntary and involuntary.

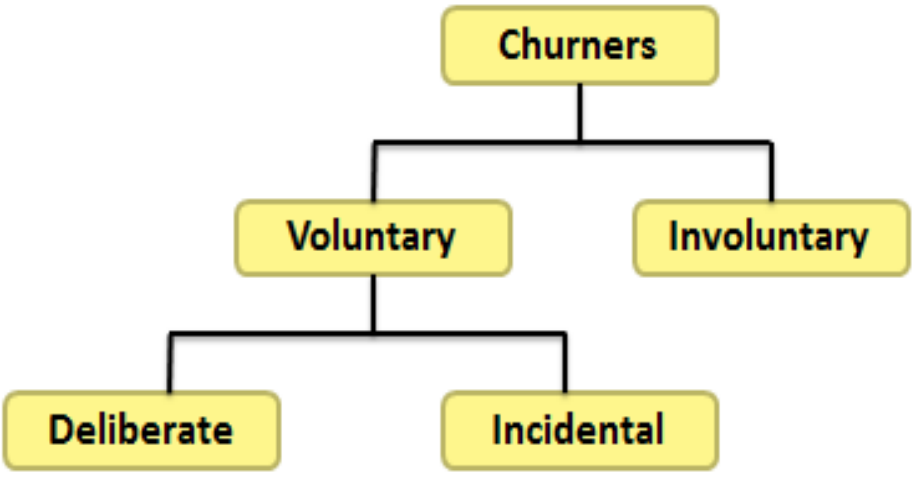

Figure 1.1: Churn categories.

Involuntary type: includes the customers that companies decide to remove from their subscribers' list. The most frequent reasons this happens is fraud cases or no payment (credit problems).

Voluntary type: includes the customers that initiate the termination of their service contract. This type of churning can be sub-divided into two new categories, incidental and deliberate. Incidental churn occurs when a customer is forced to change provider, for example, due to leaving the country. On the other hand, deliberate churn happens for reasons like technology update (customer desire newer technology), economics (price sensitivity), service quality factors, and even for social and psychological reasons.

According to the authors in [2], the cost of obtaining a new customer is five times higher than maintaining an existing customer. Therefore, most operators invest a large part of their revenues to keep up with the competition, to acquire as many customers and to expand their customer base.

1.2 Our goal

The goal of this thesis is to survey a real-world dataset that includes information about the relationship between customers and providers. Based on machine learning techniques, we combine a range of algorithms and methods and apply them on these data, trying to produce an effective and robust churn prediction model.

1.3 Chapters' organization

The rest of this thesis is organized in chapter 2 which describes related works while chapter 3 recalls the theory of machine learning algorithms used in this research. Chapter 4 provides information about the dataset and the software tools used. Chapter 5 includes details about the implementation of this model and finally in chapter 6 reported results of our prediction model.

Chapter 2

2.1 Related Work

Telecom churning is a serious issue that increasingly concerns providers and companies. Machine Learning (ML) techniques are a great solution to provide insight for this problem as it can predict with great accuracy customers' churning behaviors with great accuracy. As a result of this need, there are plenty of ML models already implemented towards this objective. In this section, we present related literature, in some of whichthe dataset in focus has also been used. This approach alloes for, sufficient comparison on the accuracy of the results provided in this thesis.

In [3] Amin et al designed an ML model that predicts customers' churning, using four different datasets. They focused on the imbalance of the data and tried six powerful oversampling techniques to deal with it - namely, Mega-trend Diffusion Function (MTDF), Synthetic Minority Oversampling Technique (SMOTE), Adaptive Synthetic Sampling approach (ADASYN), Majority Weighted Minority Oversampling Technique (MWMOTE), Immune centroids oversampling technique (ICOTE) and Couples Top-N Reverse k-Nearest Neighbor (TRkNN). In addition, they perform feature selection using the Minimal Redundancy Maximal Relevance (mRMR) technique. About the classification of the data, they applied Rough Set Theory (RST) in combination with four generation rules, which are Exhaustive Algorithm (Exh), Genetic Algorithm (Gen), Covering Algorithm (Cov) and RSES LEM2 Algorithm (LEM2). Their evaluation metrics, which are the metrics coming from the confusion matrix, present great performance. Specifically, the Gen algorithm on MTDF oversampled data provieds the best scores in all used datasets.

A different way of thinking about the prediction of customers' churn is proposed in [4]. In this work the authors consider that the features of churn and non-churn customers are close enough, a fact that induces uncertainty in the decisions of the classifier. To overcome this problem, they introduced an approach using a distance factor focusing on different distance zones (e.g., upper zone (greater distance factor's value) and lower zone (small distance

factor's value)) pertaining the estimation of certainty of the classifier. That is, when an instance is in the upper class the classifier will predict with strong confidence that this instance will churn and the reverse situation occurs when an instance is in the lower class. The used classifier is the Naïve-Bayes statistical algorithm. This study revealed that the decision maker should equally focus to propose a level of certainty to effectively predict the customer churn and non-churn with high certainty as well as with low certainty. Customers identified by the classifier with low certainty creates an uncertain situation to the provider because such customers may change their mind and may become churn from non-churn customers and vice versa.

Another interesting approach for predicting customers churn was proposed in [5]. This, study was based on Rough Set Theory which is a mathematical tool to handle imprecise and imperfect information, and also consists popular technique in Knowledge Discovery in Database (KDD) data mining. KDD rough set consists of five phases, data selection, pre-processing, transformation, mining, and evaluation. Intuitively, their model that was based on rough set theory separates the predictions into the churns or non-churns customers with great certainty, and into churners or non-churners though with a proportion of doubt, which is an idea similar to [4]. In addition, their implementation includes a non-rough set classifier, the Naive Bayes classifier. After employing both classifiers and comparing the results, the rough set classifier achieved the highest accuracy score, approximately 90%. Overall, this indicates that the rough set theory is effective in classifying customer churn compared to traditional statistical predictive approaches.

Chapter 3

Machine Learning Algorithms

Following we provide a concise description of what Machine Learning is and how the machine learning algorithms, used in throughout this thesis, work.

3.1 Machine Learning

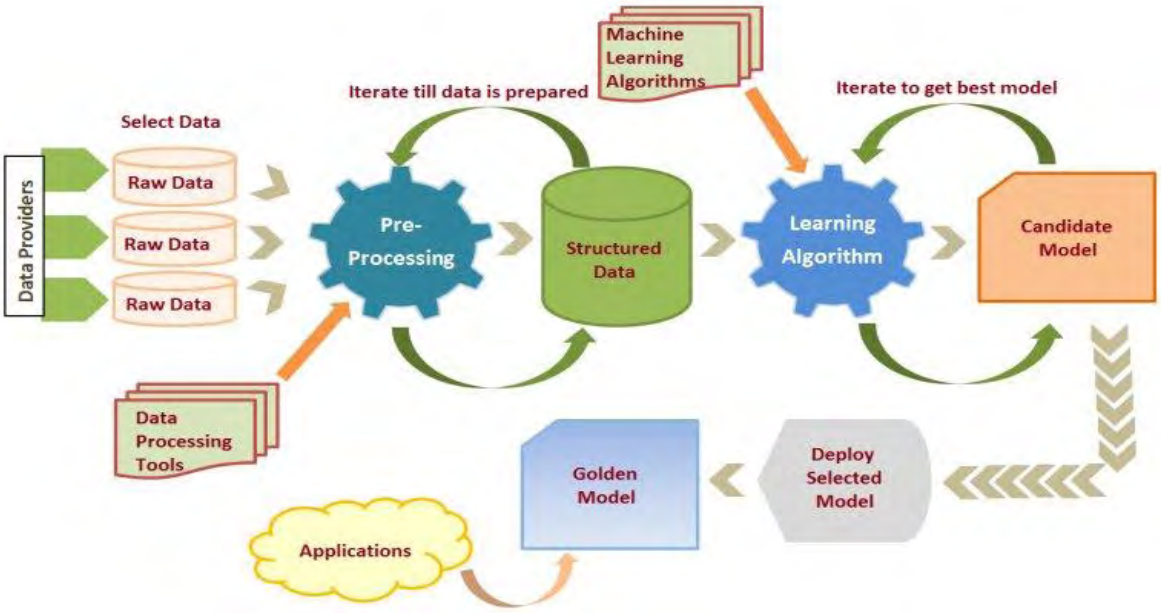

Figure 3.1: The process of Machine Learning [6].

Machine Learning is a field of Computer Science and is the procedure in which a computer program, through the continuous execution of orders improves its performance without the need for re-programming. There are plenty of definitions from great scientists, each one gives a different perspective about what Machine Learning is. Including a few of them may enhance the perception of the concept:

> *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."*

This is the first definition given by Tom Mitchell in his book *Machine Learning* at 1997 [7].

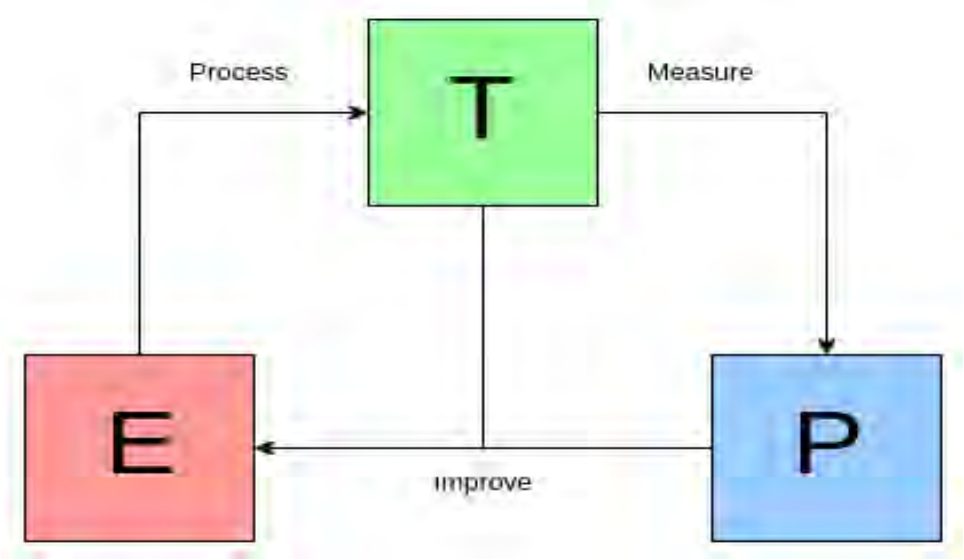

Figure 3.2: The Mitchell Paradigm [6].

Another interesting definition is given by Ian Goodfellow, Yoshua Bengio and Aaron Courville in their book *Deep Learning* where they mention:

> "*Machine learning is essentially a form of applied statistics with increased emphasis on the use of computers to statistically estimate complicated functions and a decreased emphasis on proving confidence intervals around these functions"* [8].

Drew Conway created a Venn diagram [9] to interpret the concept of Machine Learning. Figure 3.3 shows that Machine Learning is a combination of Math-Statistics and Hacking skills. D. Conway, also, has marked a danger zone on the diagram and he is referred to those people who know enough to be dangerous in the essence that they can access and structure data, run methods and present results but without understanding the meaning of these results.

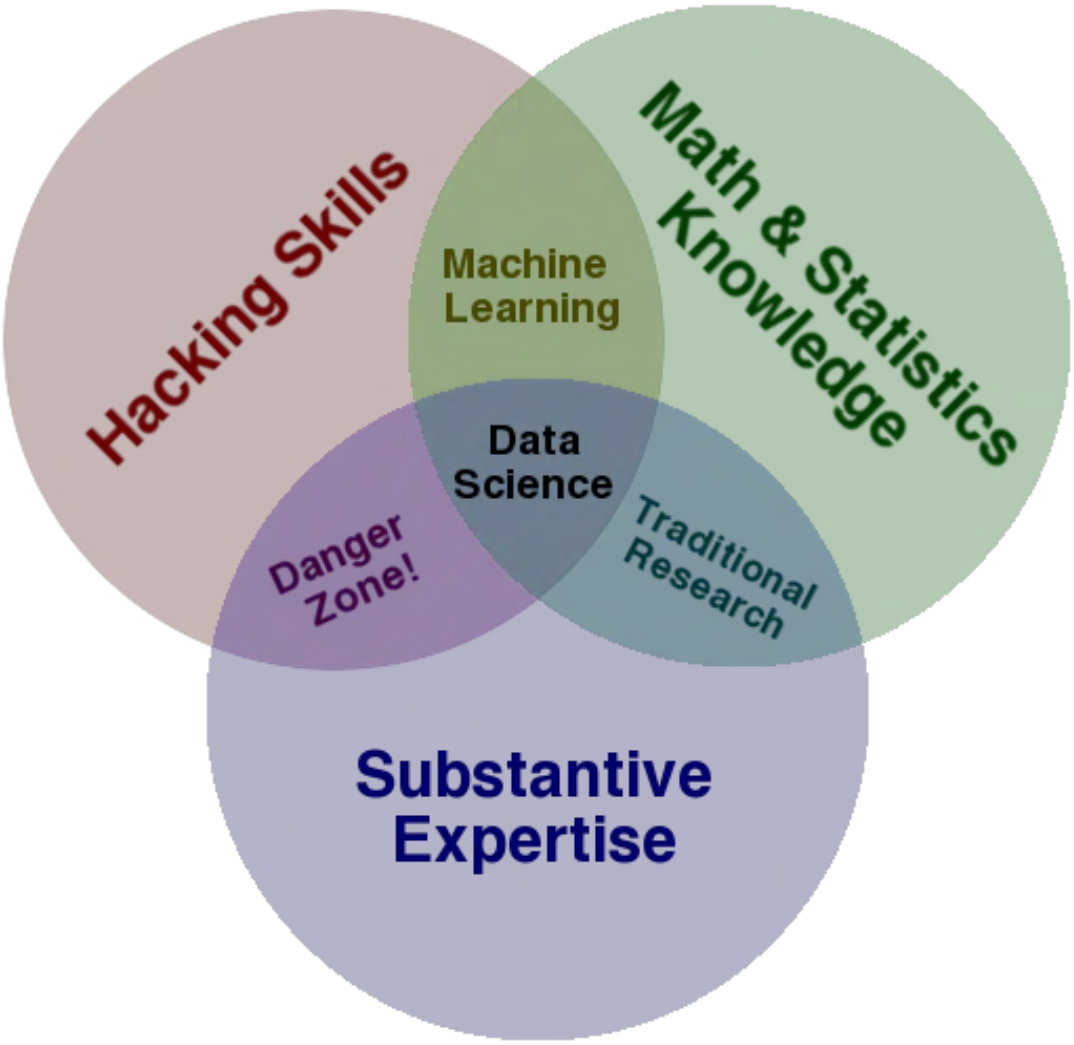

Figure 3.3: How to read data science Venn diagram [9].

Machine Learning is decomposed in three basic categories, according to the data and the way of learning from them. These are *Supervised Learning*, *Unsupervised Learning* and *Reinforcement Learning*:

- **Supervised Learning** takes place when the data contains both the input information and the target. The learning algorithm constructs a function that associates the input data to the desirable outputs, and by this way, can train effectively the model.
- **Unsupervised Learning** occurs when the data do not contain the target output. The learning algorithm tries to find structure in the data like clustering of data or points. In this way, the model that is built discovers patterns of the data and can group the inputs into clusters.

- **Reinforcement Learning** is the procedure when the algorithm can learn a strategy of actions through direct interaction with the environment. This type of learning is used in problems of a robot's motion or optimization of work in factory premises.

All the above machine learning methods are applied to a multitude of data, which for the better handling and training of the model are split into two sets. The first set is used in the training of the algorithm and is called the training set and the second is the test set, which is used for the evaluation of the model. These two sets result by splitting the initial dataset. The most common ratio for the training set is 2/3, and for the test set is the rest 1/3.

Another kind of categorization which is also usual, is associated with the type of the outcome:

- **Classification** methods are about classifying the output into two or more classes. Usually, data that are included in these methods are categorical data.
- **Regression** methods have to do with continuous values and try to predict a specific number.
- **Clustering** methods are doing almost the same thing as classification. These methods also try to classify the data but without the knowledge of the classes.

Classification and Regression methods belongs to Supervised Learning in contrast with the Clustering method which belong to Unsupervised Learning.

Based on the aforementioned, the problem of customer churn prediction that this thesis surveys, belongs to Supervised Learning and consists a Classification problem. That is, the target variable which is Churn is included in the dataset and also the kind of prediction is about in which class every customer belongs to, the Churn class or not.

3.2 K-folds Cross-Validation

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. In Machine Learning, it is applied to estimate the capability of the learning algorithm, to predict correct unseen data [10].
The way, cross-validation works depends on a single parameter k. K parameter is an integer which shows the number of groups that a given data set is going to be split.

In figure 3.4, the initial dataset is split into two sub-sets, the training and the test sets. The training set is then split again repetitively into k = 5 ‘folds’, and each fold is split again in training and test samples.

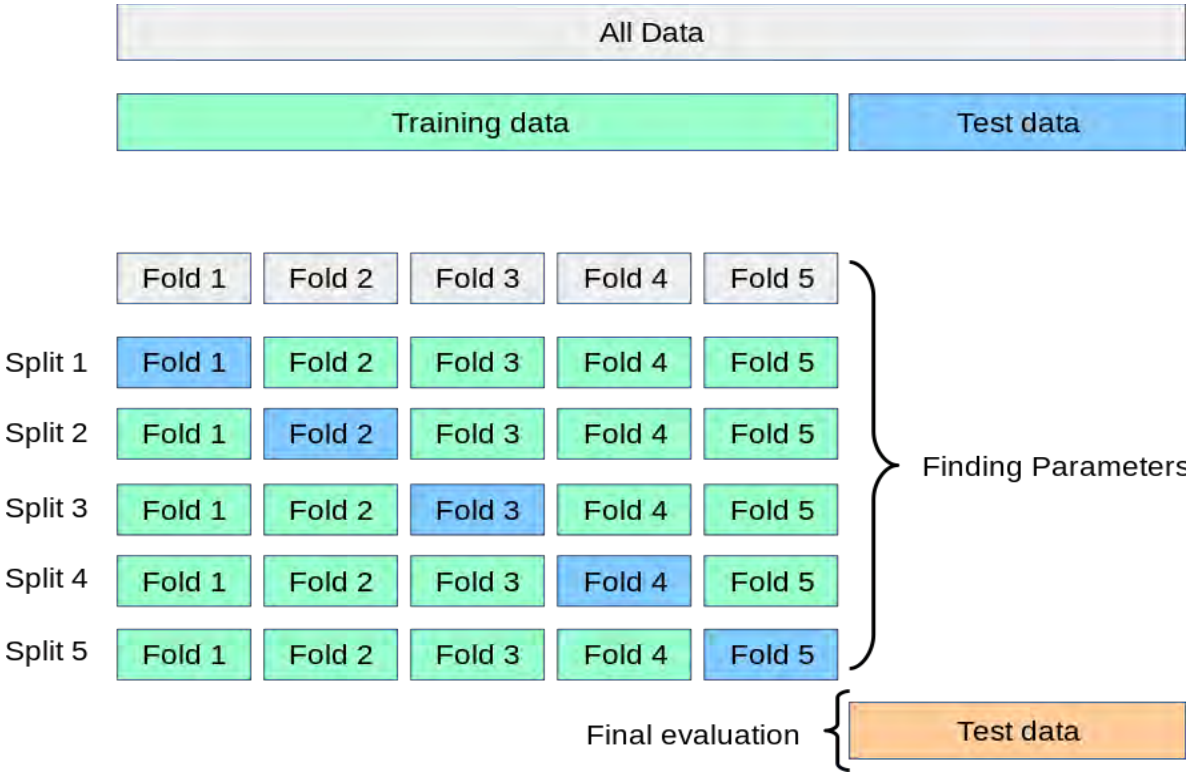

Figure 3.4: Cross-validation model [11].

Cross validation is a simple procedure to understand and implement and further make the learning model more robust and less biased.

3.3 Learning Algorithms

In this sub-section we provide the intuitive definitions of each learning algorithm used for binary classification in theaddressed problem.

3.3.1 Logistic Regression

In [12] there is a comprehensive definition about Logistic Regression (LOG_REG), which is a statistical model that its based on the logistic function, used to model a binary dependent

variable:

$$F(x) = \frac{1}{(1+e^{-x})} \quad (1)$$

Where **e** is the Euler's numerical constant and **x** is the input into the function. This function has the potential to transform every input into the range [0, 1].

In machine learning, logistic regression model takes real-valued inputs, and make a prediction regarding the probability of the input belonging to the default class, let say class 1. If the returned probability is greater than 0.5, the output is the prediction for the default class, so for class 1, else the prediction indicates the other class, class 0.

$$\text{Output} = b_0 + b_1 \cdot x_1 + \cdots + b_n \cdot x_n \quad (2)$$

The learning algorithm is responsible for discovering the most suitable coefficients (b_0, b_1, b_n). The output is transformed into a probability using the logistic function:

$$\text{P(class = 1) = } 1/((1 + e\text{^}(-output))) \quad (3)$$

Stochastic Gradient Descent is an effective procedure by which the logistic regression model can estimate the coefficients' values. The concept is to predict a possible output for each instance in the training set and calculate the error for each prediction. This process is repeated until the error converge in a small number or for a fixed number of iterations.

$$b = b + \alpha \cdot (y - prediction) \cdot predictiion \cdot (1 - prediction) \cdot x \quad (4)$$

Where a is the learning rate and controls how much the coefficients (and therefore the model) changes or learn each time it is updated.

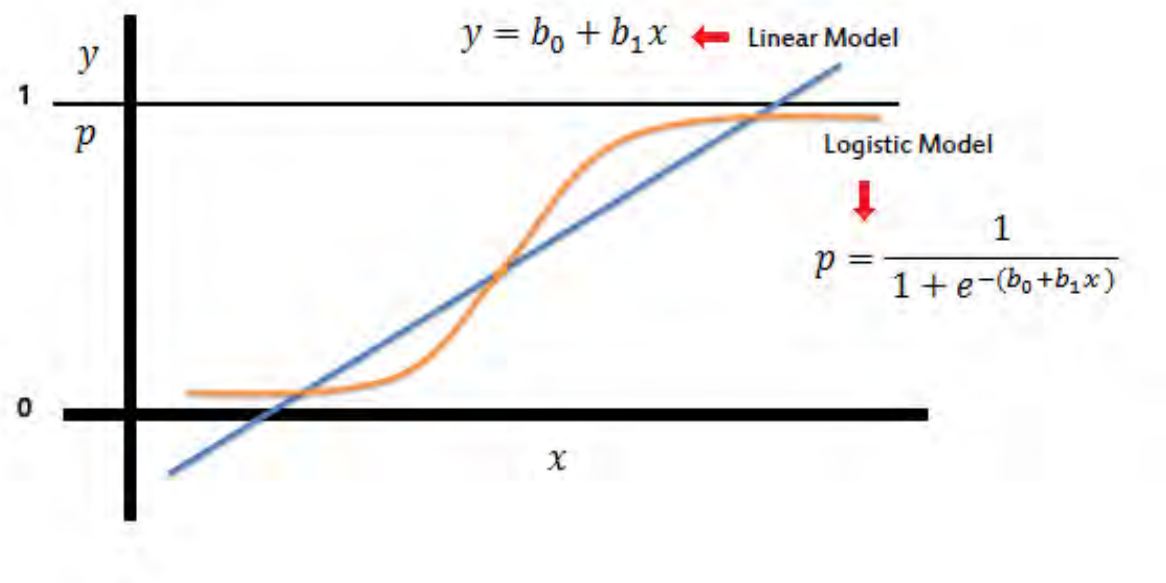

Figure 3.5: Logistic Regression model [13].

3.3.2 K – Nearest Neighbors

K Nearest Neighbors (KNN) algorithm stores all available instances in the training set and classifies new input data based on a similarity measure, like distance functions. KNN is used from 1970 in statistical estimation and pattern recognition [13].

The objective of this algorithm is to classify a new instance taking into consideration the majority vote of its neighbors. The class, in which this instant should be placed, is the class where most of its K nearest neighbors have been placed, calculated by a measure function. The three distance measures that are used to determine which of the K neighbors are most similar to a new input are:

- *Euclidean distance*: $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$ (5)

- *Manhattan distance*: $\sum_{i=1}^{k}|x_i - y_i|$ (6)

- *Minkowski distance*: $\left[\sum_{i=1}^{k}(|x_i - y_i|)^q\right]^{1/q}$ (7)

It is important that all of these functions are valid for continuous values.

Below, figure 3. 6 depicts briefly the KNN algorithm..

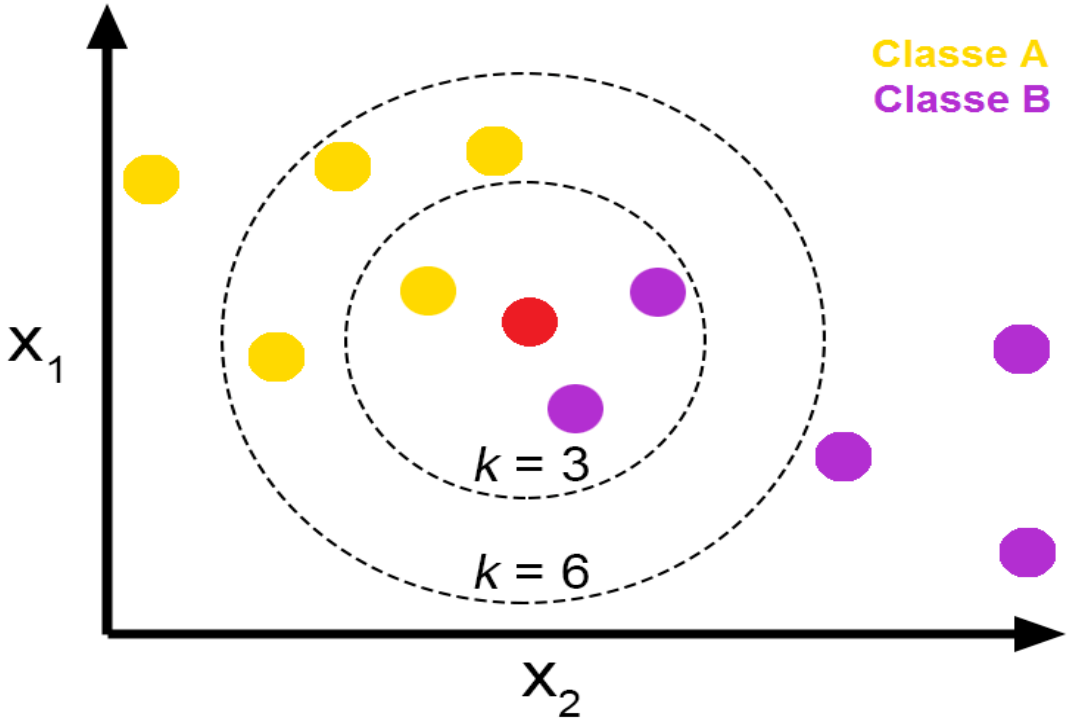

Figure 3.6: KNN model [14].

3.3.3 Support Vector Machine

Support Vector Machine (SVM) was invented in 1963 by Vladimir N. Vapnik and Alexey Ya. Chervonenkis and became quite known as a method around 1990 [15]. Initially this method had begun to be used only for classification problems but subsequently developed SVM models were used in troubleshooting of regression problems.

SVM based algorithms used to separate data into two categories or classes. Their basic goal is to create a line or a hyperplane, that splits the two classes, and maximize the margin,

between the two classes. This margin is the distance between the line or the hyperplane and the closest data points. The best or optimal line that can separate the two classes is the line that has the largest margin [15]. This is called Maximal – Margin hyperplane. The way it is measured is by taking the perpendicular distance from the line to the closest points. These points are called support vectors, and support or define the line or the hyperplane.

The line or decision boundary separates the two classes is the: $wx + b = 0$. Moreover, taken the $wx + b = -1$ and the $wx + b = 1$, the perpendicular distance between those vectors and the hyperplane is:

$$d = \frac{2}{\|w\|} \qquad (8)$$

Thus to achieve our goal, maximize the distance *d*, we should minimize the equation (8).

$$min \frac{1}{2} \|w\|^2 \quad , \quad y_i \cdot (w \cdot x_i + b) \geq 1, \forall x_i \qquad (9)$$

The above formula (9) could be applied to an ideal sorting problem where all items would be completely classified, which is unseen in real-world data. The constraint of maximizing the margin of the line that separates the classes must be relaxed. This is often called the soft margin classifier. In this way, the desired hyperplane is a softer surface $\boldsymbol{\xi_i}$ making as fewer misclassification mistakes as possible. Also, there is a tuning parameter C that defines the amount of violation of the margin allowed. A C = 0 is no violation and the result is the inflexible Maximal-Margin classifier. The larger the value of C the more violations of the hyperplane permitted.

The final formula is :

$$m_{in} \left[\frac{(\|w\|)^2}{2} + C \sum_{i=1}^{N} \boldsymbol{\xi_i} \right] , y_i \cdot (w \cdot x_i + b) \geq 1 - \xi_i , \forall x_i, \boldsymbol{\xi_i} \geq 0 \qquad (10)$$

Figure 3.7 sums up all the above. The left plot shows that there are many lines in 2D that can separate the two classes, b ut the right plot depicts the optimal line that separates these classes.

The SVM algorithm is implemented in practice using a kernel. Depending on the kernel SVM, can deal with linear problems as well as with nonlinear. The most popular are:

- *Linear kernel*: $k(x_i, y_i) = x_i y_i + \gamma$ (11)
- *Polynomial kernel:* $k(x_i, y_i) = (x_i y_i + 1)^{\rho}$ (12)
- *Radial kernel:* $k(x_i, y_i) = e^{-\gamma(x_i - x_j)^2}$ (13)
- *Gaussian kernel:* $k(x_i, y_i) = e^{\frac{-1}{2\sigma^2}(x_i - y_1)^2}$ (14)

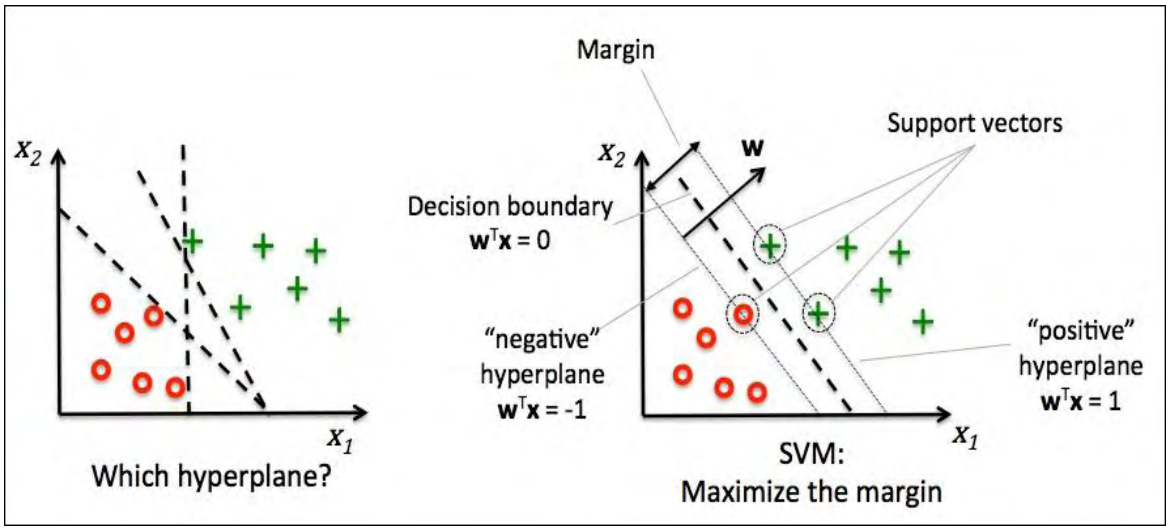

Figure 3.7 : SVM and Maximal margin [16].

3.3.4 Naive Bayesian

As mentioned in [17], naïve Bayes classifier belongs to a family of probabilistic classifiers, which is based on the application of Bayes theorem with strong or naïve independence assumptions between the features. First of all, the formula of Bayesian theorem is:

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)} \quad (15)$$

Where:

P(c|x) is the posterior probability of class (target) given predictor (attribute)

P(c) is the prior probability of class

P(x|c) is the likelihood which is the probability of predictor given class.

P(x) is the prior probability of predictor.

The naïve Bayesian classifier bases its predictions on this formula. After calculating the posterior probability for a number of different classes, selects the class with the highest score. This is the maximum probable class (MAP):

$$MAP(c) = max\big(P(c|x)\big) \qquad (16)$$

3.3.5 Ensemble Methods (Random Forest, XGBoost)

Ensemble methods are another popular machine learning technique. Their popularity is due to the fact that they succeed in a great performance in almost every kind of problem they are used to solve. Specifically, the idea behind this technique is to combine many learning algorithms or models to predict the target output. There are plenty of definitions, trying to explain this concept. An interesting one given by the data scientist, Vadim Smolyakov, in [18] says that ensemble methods are meta-algorithms that combine machine learning techniques into one predictive model in order to decrease variance (bagging), bias (boosting), or improve predictions (stacking).

There are many kinds of ensemble techniques and they are usually separated as follows:

- Basic Ensemble Techniques:
    - Max Voting
    - Averaging
    - Weighted Average
- Advanced Ensemble Techniques
    - Stacking
    - Blending
    - Bagging

- Boosting

After this separation, we focus on two of the advanced ensemble techniques, Bagging and Boosting.

Bagging is a combination of words Bootstrap and Aggregation. Bootstrap is a sampling technique, that creates subsets of instances from the original dataset with replacement. Subsequently, this method feeds these subsets or bags to the learning models and aggregates the results to each of these learners to predict the target.
A very well-known algorithm, which belongs in to the Bagging category is **Random Forest**. Random Forest is an ensemble learning algorithm for both regression and classification problems, that implement exactly the strategy of bagging. In particular, constructs in parallel plenty of decision trees, gives them bootstrapping samples and the outcome is the mode of the classes in classification problems or the mean in regression problems.

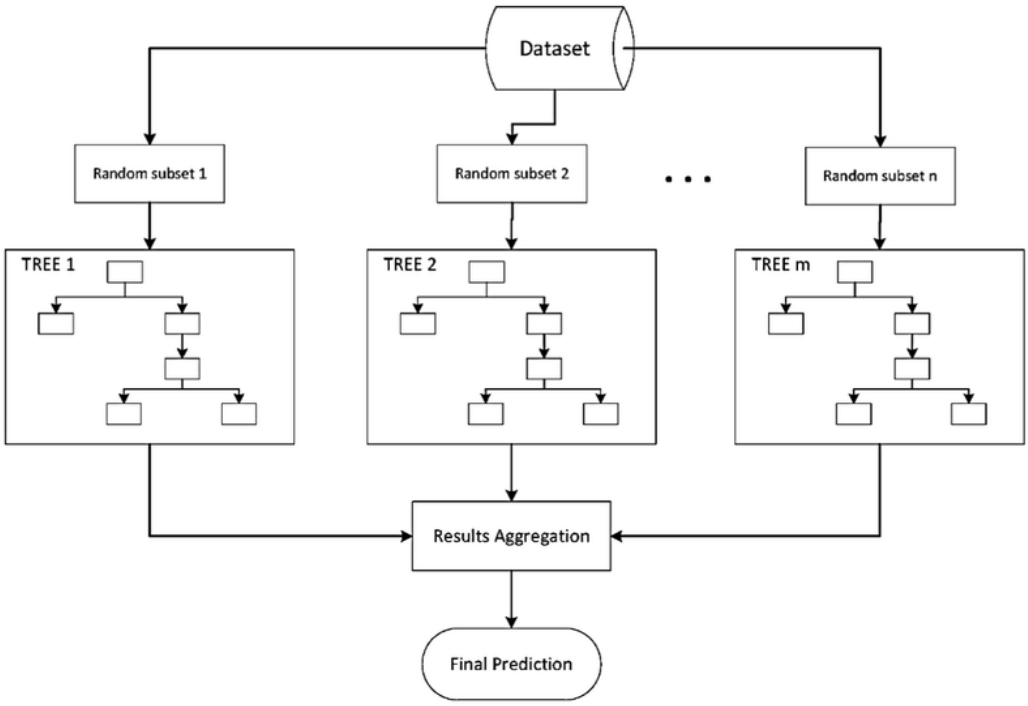

Figure 3.8: Depicts the Random Forest algorithm [19].

On the other hand, **Boosting** methods are able to convert a weak learner to a strong learner. A classifier learning algorithm is said to be weak when small changes in data induce big changes in the classification model. This technique is about a sequential model, that starts

with a weak leaner, usually a decision tree, make a prediction and calculate how far hat prediction was from the target, or simply the error which is also called residual. Next step is to construct another decision tree but this time the target is the reduction of the residual from the previous learner, and this procedure goes on until the error converges in a small number. An example of how boosting algorithms work is depicted in figure 3.9. Looking at the upper left figure, there is a split between blue and the orange items. The next step is to catch the blue miss-classified items so in the next figure, the split is moved. In the upper right figure, this time the algorithm tries to catch the orange items correctly. Finally, in the bottom figure, the model sums all the previous splits and learns how to split the items in the right way.

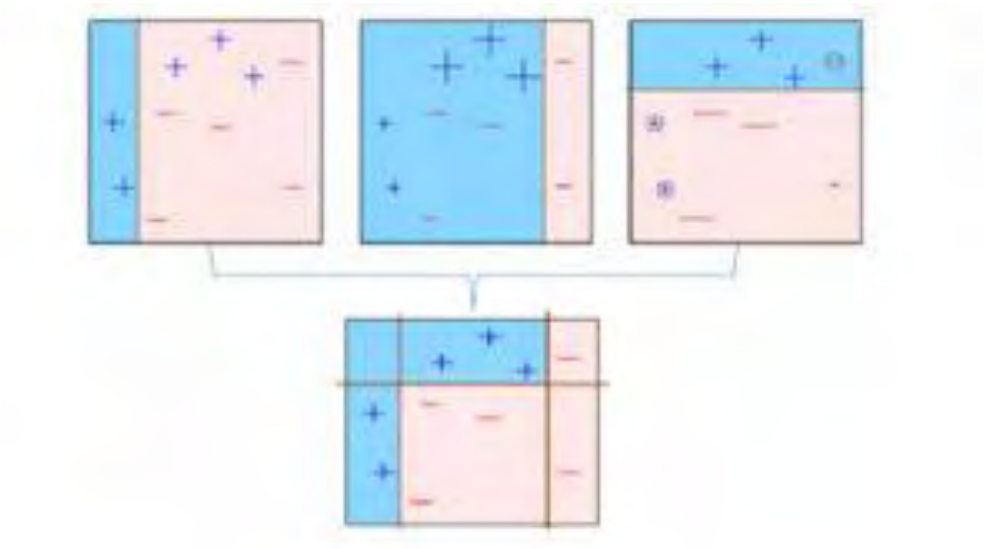

Figure 3.9: How boosting algorithms works [20].

The algorithm, that we used in our case, comes from this family of boosting methods and is titled **Extreme Gradient Boost** (**XGB**). Tianqi Chen, and Carlos Guestrin presented their idea about XGBoost algorithm in [21]. XGBoost that was created by Tianqi Chen, is an implementation of gradient boosting machines. More specifically, is a software library designed and optimized for boosted tree algorithms. The main goal is to push the extreme of the computation limits of machines to provide a scalable, portable and accurate algorithm for large scale tree boosting.

3.4 Evaluation Metrics and Criteria

This section provides the metrics that are used to evaluate the performance of the models. We also analyze and explain where we focus on and why.

3.4.1 Confusion Matrix

Confusion Matrix is a table that describes the models' effectiveness and performance, on test subset where the true values are unknown. Its structure seemed in table 3.1.

- **True Positives (TP):** when the actual class of the data point was 1 (True) and the predicted is also 1 (True).
- **True Negatives (TN):** when the actual class of the data point was 0 (False) and the predicted is also 0 (False).
- **False Positives (FP):** when the actual class of the data point was 0 (False) and the predicted is 1 (True).
- **False Negatives (FN):** when the actual class of the data point was 1 (True) and the predicted is 0 (False).

Accuracy is the percentage of total items classified correctly:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (17)$$

Recall or Sensitivity or TPR is the number of items correctly identified as positive out of total true positives:

$$Recall = \frac{TP}{TP+TN} \quad (18)$$

Precision is the number of items correctly identified as positive out of total items identified as positive:

$$Precision = \frac{TP}{TP+FP} \quad (19)$$

Specificity is the number of items correctly identified as negative out of total negatives:

$$Specificity = \frac{TN}{TN+FP} \quad (20)$$

		Predicted		
		Non-Churn customer (Class 0)	**Churn customer (Class1)**	**TOTAL**
Actual	**Non-Churn customer (Class 0)**	**TN (True Negative)**	**FP (False Positive)**	**Actual Positive Number / Specificity**
	Churn customer (Class 1)	**FN (False Negative)**	**TP (True Positive)**	**Actual Positive Number / Recall**
	TOTAL	**Predicted Negative Number**	**Predicted Positive Number / Precision**	**Total Customer Number**

Table 3.1: Confusion Matrix.

A closer look at the above equations and table 3.1, shows that there is a trade-off between the precision and recall metrics. Intuitively, recall gives information about how many items, which were actually true, the classifier missed. On the other hand, precision calculates how many items, that were actually true, predicted as true. When the recall is increased, the precision is decreased and the reverse. It depends on the kind of problem which one of the two metrics does matter more. In this survey, we want to find as more customers that are going to churn as possible, and for this reason, we should minimize the False Negatives. We consider that is more important for the providers to know how many clients are going to churn, including a tolerable percentage of customers that will not, rather than miss these clients in favor of precision.

It would be difficult to compare both metrics for each model and choose the best. That is, **F1 score** is a single number evaluation metric that combines both recall and precision. More specifically, the F1 score is the harmonic mean between recall and precision:

$$F1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} \tag{21}$$

In this way, calculating *F1 score* for each model or classifier, we get a single score which represents both precision and recall and is easier to make comparison between them.

3.4.2 AUC score and ROC curve

Two additional reliable metrics of models' performance, which are usually going together, are the Area Under the Curve (AUC) and Receiver Operating Characteristic (ROC) curve. Particularly, ROC is a probability curve and AUC represents the degree or measure of separability. The inference that is drawn from AUC is about how well the classifier distinguishes the two classes. The higher the AUC the better the model predicts 0s as 0s and 1s as 1s. Below, figure 3.10 depicts some ROC curves between the ideal classifier and the random one. In addition to this, it is an effective way to compare the performance of many models, by drawing, in the same plot, the ROC curves of each model.

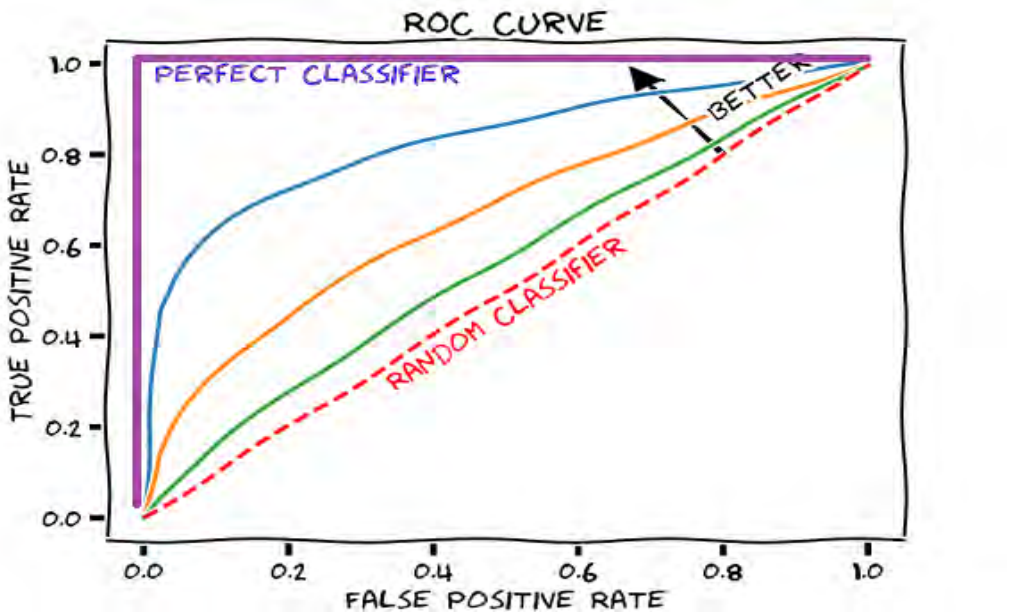

Figure 3.10: Roc curves [22].

Chapter 4

Dataset - Tools

4.1 The Dataset

The dataset we used in this survey was taken from Kaggle. Kaggle was founded by Anthony Goldbloom and Ben Hammer in April 2010. As mentioned in Wikipedia [23] is an online community for data scientists and machine learners, which can explore and build models in a web-based data-science environment work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

This IBM Sample Dataset [24] has information about Telco customers, a history record, and their decisions to leave or remain in the company. More specifically, there are 7043 rows that correspond to the customers and 21 columns that correspond to the customers' attributes. A detailed description of the dataset's variables is probided in Table 4.1. In addition, an intuitive understanding of features could be by categorizing them as follows:

- *Customers who left within the last month* – the column is called Churn.
- *Services that each customer has signed up for* – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies.
- *Customer account information* – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges.
- *Demographic info* about customers – gender, age range, and if they have partners and dependents.

Feature Name	Type	Description
customerId	alphanumeric	Customer ID
gender	String	Whether the customer is Male or Female
SeniorCitizen	Number	Whether the customer is a senior citizen or not (0,1)
Partner	String	Whether the customer has a partner or not (Yes, No)
Dependents	String	Whether the customer has dependents or not (Yes, No)
Tenure	Number	Number of months the customer has stayed with the company
PhoneService	String	Whether the customer has a phone service or not (Yes, No, phone service)
MultipleLines	String	Whether the customer has multiple lines or not (Yes, No, phone service)
InternetService	String	Customer's internet service provider (DSL, Fiber optic, No)
OnlineSecurity	String	Whether the customer has online security or not (Yes, No, No internet service)
OnlineBackup	String	Whether the customer has online backup or not (Yes, No, No internet service)
DeviceProtection	String	Whether the customer has device protection or not (Yes, No, No internet service)
TechSupport	String	Whether the customer has tech support or not (Yes, No, No internet service)
StreamingTV	String	Whether the customer has streaming TV or not (Yes, No, No internet service)
StreamingMovies	String	Whether the customer has streaming movies or not (Yes, No, No internet service)
Contract	String	The contract term of the customer (Month-to-month, One year, Two year)
PaperlessBilling	String	Whether the customer has paperless billing or not (Yes, No)
PaymentMethod	String	The customer's payment method (Electronic check, Mailed check, Bank transfer(automatic), Credit card(automatic))
MonthlyCharges	Number	The amount charged to the customer monthly
TotalCharges	String	The total amount charged to the customer
Churn	String	Whether the customer churned or not (Yes or No)

Table 4.1: Description of features in the dataset

4.2 Tools

This section contains a description of the working environment as well as the tools used to carry out the study.

4.2.1 Python

The whole code of this project has been written in Python. The version that was used was Python 3.6.8. Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented

programming [25]. In the last few years, it seems to be one of the most popular programming languages, especially in data science and data analytics. This happens because it includes some powerful libraries for handling data structures, like *Pandas* or *Numpy* for math computations. In addition to these, there are also some useful plot libraries like *Matplotlib*.

4.2.2 Anaconda

Anaconda is a free platform that includes many useful and powerful packets for mathematics, science and engineering already installed [26]. Also, it gives to users the ability to make an easy installation of the desirable version of python.

4.2.3 Jupyter Notebook

One of the greatest tools provided with Anaconda installation is *Jupyter Notebook*. Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text [27]. Moreover, the notebook has support for over 40 programming languages, including Python, R, and Scala. The whole project implemented in Jupyter Notebook resulting in rich interactive output like HTML files, images and plots.

Chapter 5

Implementation

5.1 Pre-processing the data

Machine Learning (ML) is all about mathematical and statistical models with equations and numerical operations. That is, the input data to these algorithms should be numbers so they work in a proper way. The first level of the implementation is about the preparation of the dataset.

Firstly, using *Pandas* python's library, we clean the data by non-appropriate or duplicated values. Also, there are useful Panda's functions that return descriptive statistics and information about the variables of the dataset.

At this point, in order to achieve a more direct and easier understanding of the data and their importance, we have constructed some plots. In more detail, we split the features in categorical and numerical. In figure 5.1, are depicted the percentages of customers that churn or not, against a specific categorical variable each time. There are sixteen bar plots corresponding to the sixteen categorical variables of the dataset, except the Churn which is the target variable. It seems that some cases are really annoying for the customers, which lead them to churn. Such cases are when the payment method is the Electronic check option, or when Internet service online security is not applied. A remarkable case is that clients churn if there is the option of the Fiber optic internet service.

The same process was followed for constructing the plots in figure 5.2. This time, we used density plots because we handled the numerical attributes and we wanted to reveal their distributions. The left plot shows the correlation between the tenure variable with the target Churn variable and intuitively we could infer that recent customers are more possible to churn than the customers who are long time customers of the specific provider. In a similar way both remaining plots, the middle and the right one, depict the correlation between TotalCharges and Churn, and MonthlyCharges and Churn respectively. Intuitively, we could

conclude that customers with high monthly charges are possible to churn. On the other hand, the plot about total charges and churn customers does not provide a useful insight.

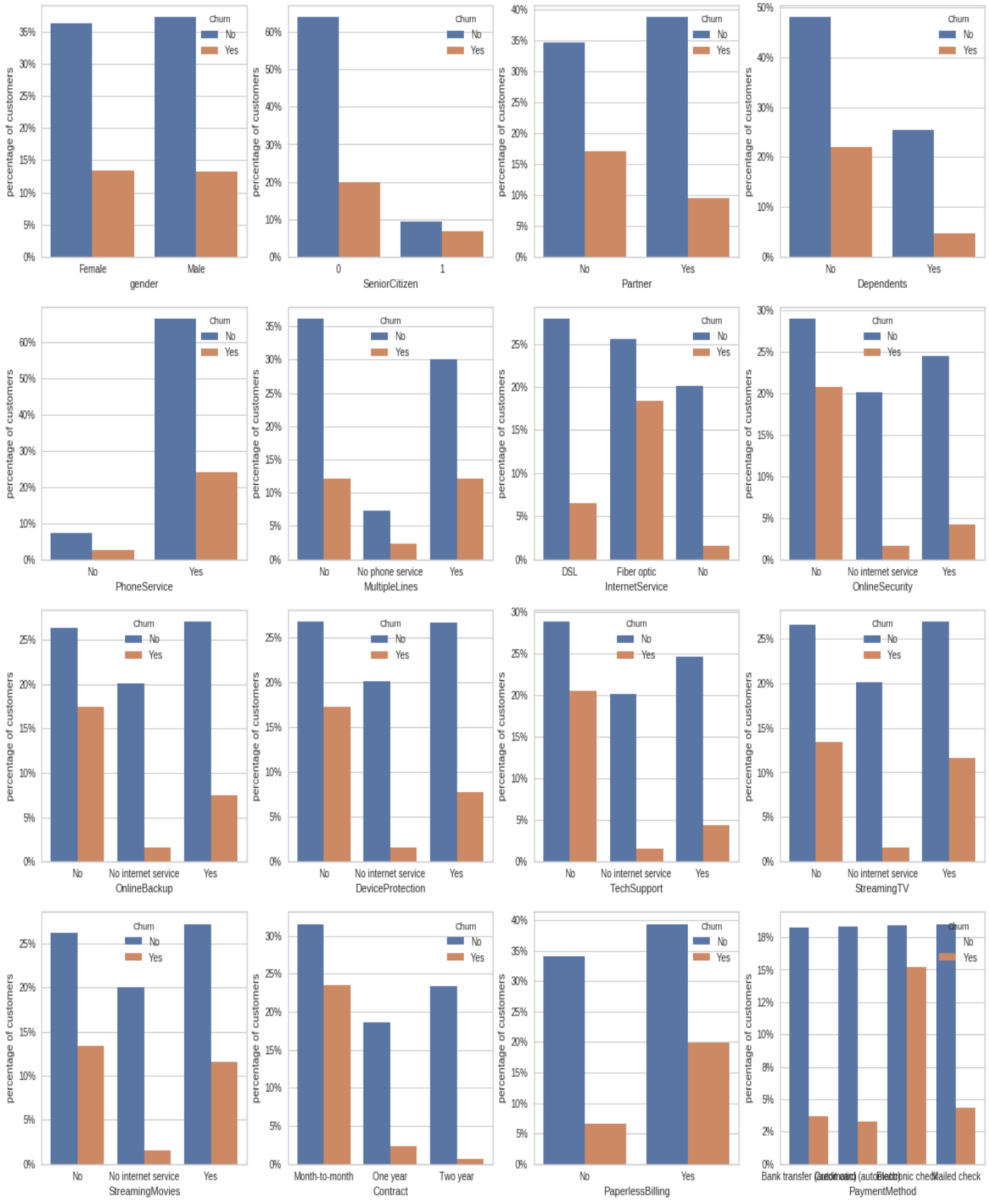

Figure 5.1: Bar plots of categorical data against Chun variable.

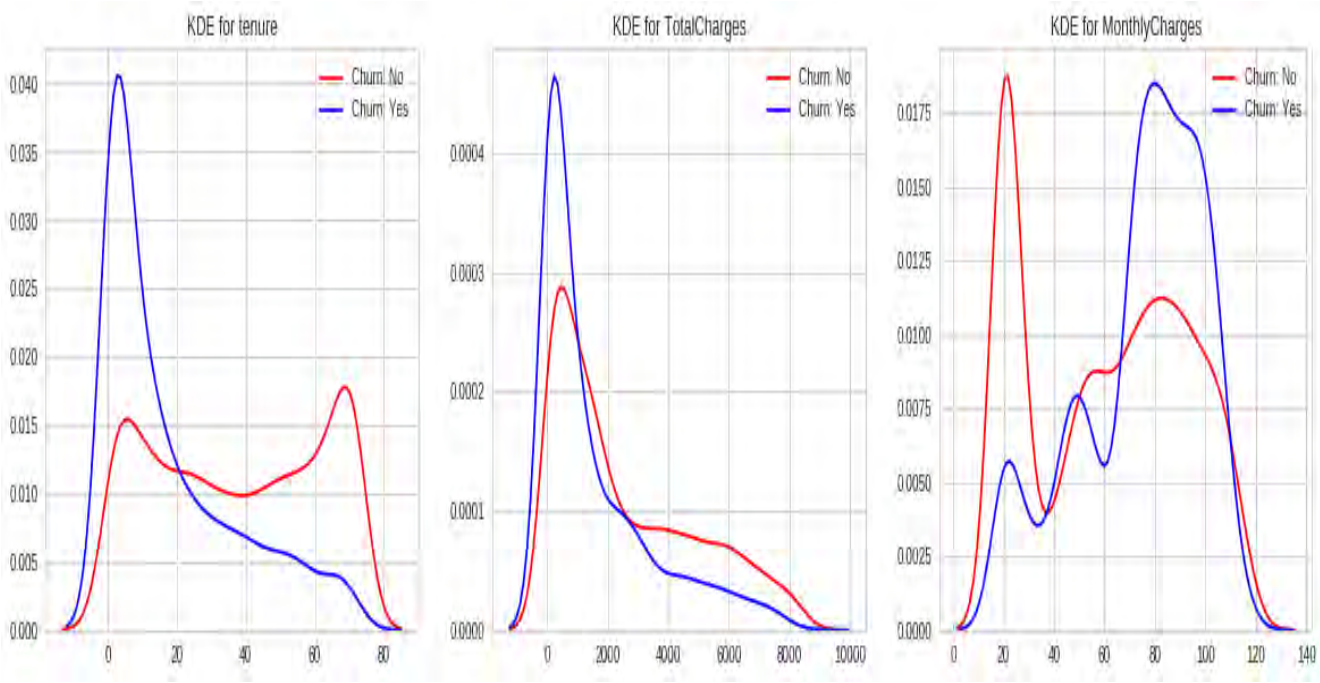

Figure 5.2: Density plots of numerical variables against churn variable.

Next, we discard the feature "*customerID*", because if a customer is willing to churn, it has no correlation with his/her ID. As seen in table 4.1 of features, other than the three variables which are "*SeniorCitizen*", "*tenure*" and "*MonthlyCharges*", the rest are categorical variables and it is necessary to convert them into numbers.

Using the *One Hot Encoding* we solved this problem. One Hot Encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction [28]. In particular, this method performs binarization of the categorical features as this is convenient for the dataset we used. Most of the variables have a range of 2 or 3 values, so the binarization is helpful. Among that, the columns of the dataset are increased from 21 to 41 because after the implementation of One-hot encoding every categorical feature (one column) now corresponds to 2 or 3 columns (it depends on the range of values that each feature has).

In addition, we scaled all the data except for those that came from One-hot encoding. The scaling was implemented with *StandardScaler*, a method that is included in sickit-learn

python's library [29]. In this way, we standardized the features by removing the mean and scaling them to unit variance.

After scaling the data, we split the independent variables from the dependent one. As known, the dependent variable in our dataset is the "*Churn*" feature. Meanwhile, by plotting this variable in figure 5.3, we identify that the dataset is imbalanced. More specifically, 73.5% of all the customers in the whole dataset will not churn whilst 26.5% of them will do.

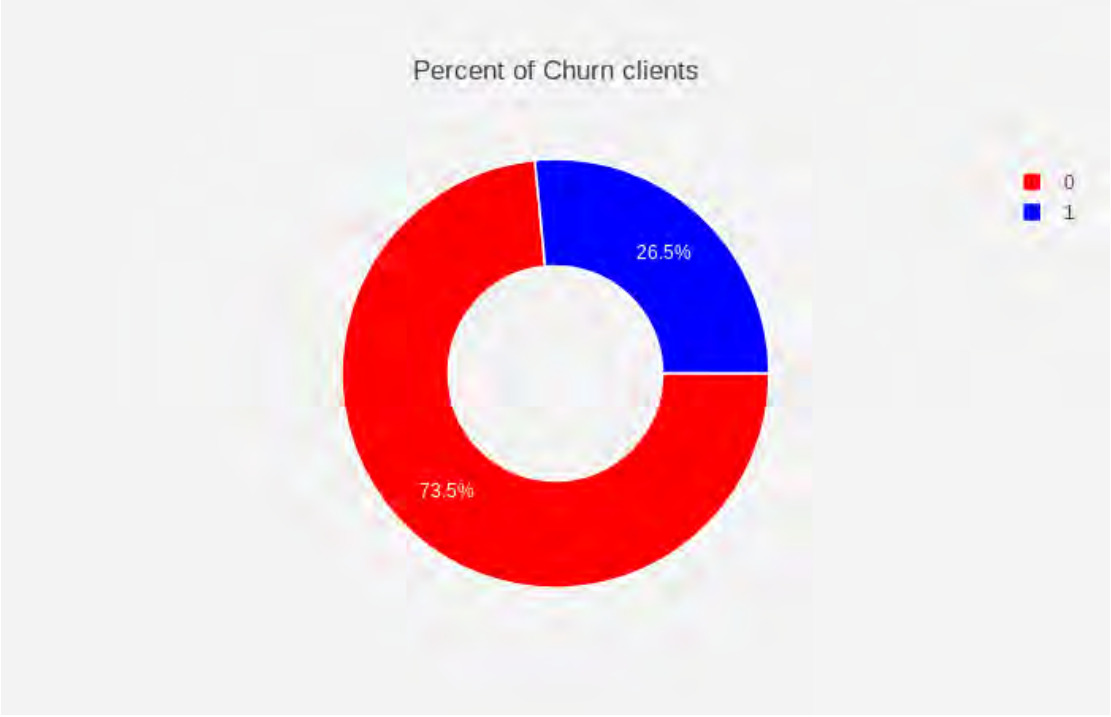

Figure 5.3: Pie chart percentages of customers churn or not.

5.2 Imbalanced Data

As it seems in figure 5.3, the distribution of the target variable in dataset, *Churn,* is skewed. Specifically, the percentage of churners is 26.5% against 73.5% of non-churners. The bad evaluation scores that the algorithms achieve as shown below, seems to be due to this imbalance on dependent variable. In addition, there is the paradox of accuracy metric, because the classifier is possible to predict the majority class against the minority one and

achieve a high score. In reality, we can not predict it in a proper way. Jason Brownlee refers in [30] that *"it is the case where your accuracy measures tell the story that you have excellent accuracy (such as 90%), but the accuracy is only reflecting the underlying class distribution"*.

Additionally, Brownlee [30] suggests 8 ways of how to deal with the imbalance learning. In this study, we perform the resampling method to handle the imbalanced data. Resampling methods could be split into oversampling and undersampling methods. The fact that the dataset we working on is not that big, has 7043 instances, urges us to choose the oversampling technique. One the other hand, if we used undersampling techniques, the already small dataset would be even smaller and there would not be satisfying number of data for the training of the classifier.

There are plenty of algorithms that implement oversampling technique like Random oversampling, Synthetic Minority Over-Sampling Technique (SMOTE), Adaptive Synthetic Sampling Approach (ADASYN), Mega-Trend Diffusion Function (MTDF),
Majority Weighted Minority Oversampling Technique (MWMOTE), Immune centroids oversampling technique (ICOTE) and many others. In this study, we applied two of them, the SMOTE and ADASYN.

SMOTE method was proposed in 2002 [31]. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen [32][33]. Below, figure 5.4 shows this procedure.

ADASYN works in a similar manner as SMOTE, however, it attempts to find which points in the minority class would be the most difficult for a model to learn and attempts to place a higher ratio of synthetic data close to these points.

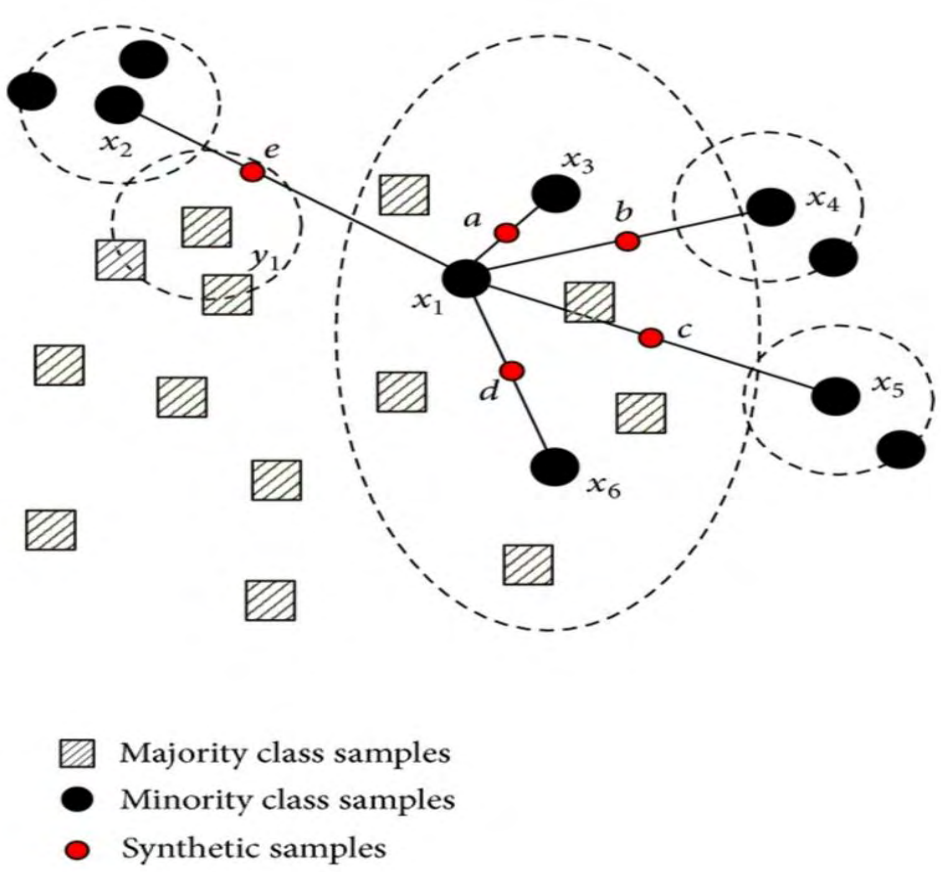

Figure 5.4: SMOTE technique [32].

In figure 5.5, after feature reduction we keep the two most valuable variables and in this way we can plot in 2D (i) the format the datasets take after SMOTE resampling (middle plot), (ii) ADASYN resampling (right plot), and (iii) without resampling (left plot).

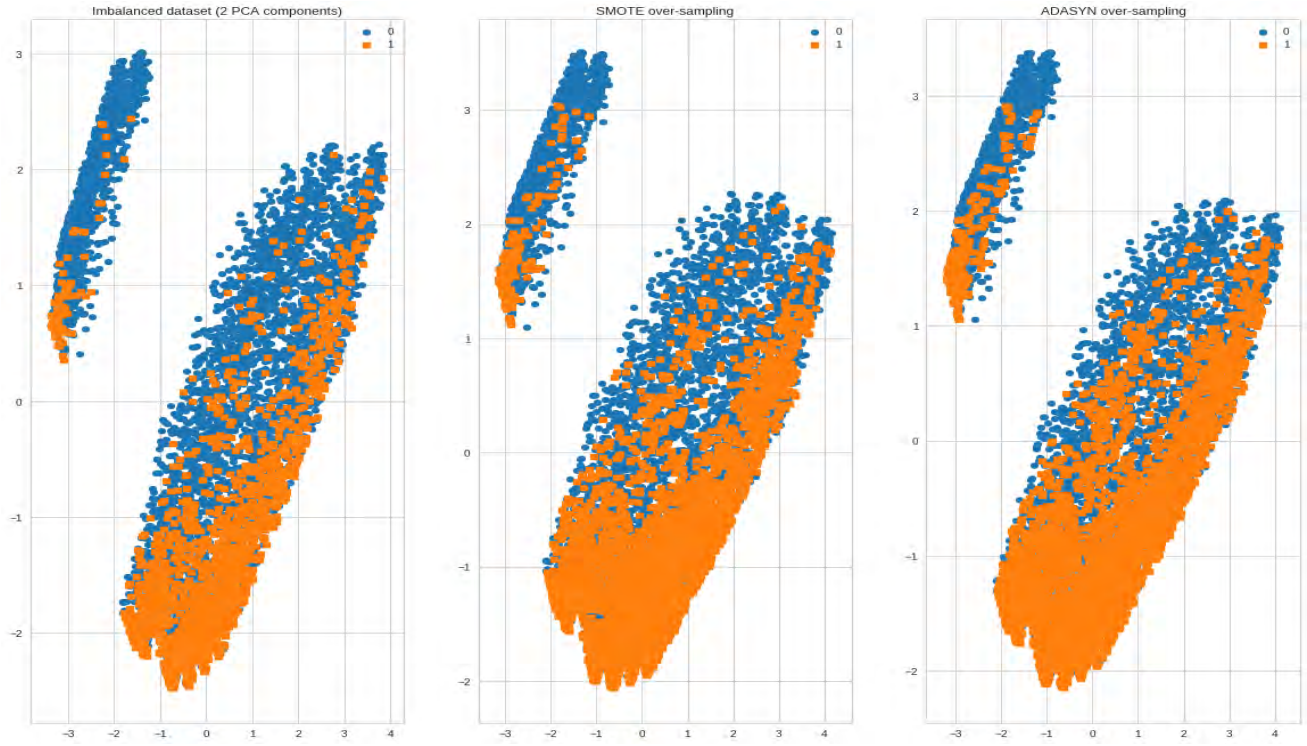

Figure 5.5: The plot in the left is the imbalanced dataset, the plot in the middle is the dataset after SMOTE technique and the right plot is after ADASYN technique.

5.3 Feature Selecting and Feature Importance

Feature selecting is the procedure in Machine Learning and statistics, where a subset of the whole dataset is chosen, to be used in the training and testing parts of the learning algorithm. This technique is considered an essential ingredient for every machine learning model because it benefits them in many ways. In [34] the four most important techniques are provided, which are the simplification of the models to make it easier to interpret by researchers, the shorter training times, the avoidance of curse dimensionality and the reduction of overfitting. The manner that thesis attribute selection algorithms work is a combination of searching for proposing the new subset of features, along with an evaluation metric that indicates the most suitable for the model subset of features. The choice of the evaluation metric heavily influences the algorithm, and it is these evaluation metrics which distinguish between the three main categories of feature selection algorithms: wrappers, filters and embedded methods.

- **Filter Methods** apply a statistical measure to compute a score for each feature, and in this way to filter and take only the subset with the most relevant features. Specifically, these scores create a ranking in which the

selecting algorithm is based to remove a feature from the dataset or to keep it. Usually, these methods are univariate and consider the feature independently or with regard to the independent variable.

- **Wrapper Methods** choose the most effective subset of features by using a machine learning algorithm. The metric for these methods is the error rate this predictive model succeeds using different combinations of subsets. The lower the better. Although wrapper algorithms produce better performing feature sets in comparison with filter methods, they are often computationally expensive and time-consuming.

- **Embedded Methods** learn which features best contribute to the accuracy of the model while the model is being created. They take care of each iteration of the model training process and carefully extract those features which contribute the most to the training for a particular iteration.

In this thesis, we try a method of each of the above categories. In particular, the first feature selecting technique we implement is the univariate selection, belonging to filter methods. Univariate selection examines each feature separately to decide whether or not there is a significant correlation with the target variable. The statistical measure is the Chi-Square Test that examines the relationship between each of the independent variables with the dependent one, by the formula:

$$x^2 = \sum_{i=1}^{n} \frac{(o_i - E_i)^2}{E_i}$$

Where O_i is the number of observations in class I, and E_i is the number of expected observations in class I if there was no relationship between the feature and target.
The greater the x^2 the better the relationship between the independent and dependent variables. In figure 5.6 we choose the 30 most significant features out of 40 in total. As seen, in the top places are located features which were expected to see, as in early preprocessing

step we had concluded in these variables, like tenure, MonthlyCharges, or PaymentMethod_Electronic_check.

	Specs	Score
8	TotalCharges	629275.029414
4	tenure	16278.923685
7	MonthlyCharges	3680.787699
33	Contract_Month-to-month	519.895311
35	Contract_Two year	488.578090
38	PaymentMethod_Electronic check	426.422767
15	OnlineSecurity_No	416.182917
24	TechSupport_No	406.117093
10	InternetService_Fiber optic	374.476216
11	InternetService_No	286.520193
16	OnlineSecurity_No internet service	286.520193
19	OnlineBackup_No internet service	286.520193
22	DeviceProtection_No internet service	286.520193
25	TechSupport_No internet service	286.520193
28	StreamingTV_No internet service	286.520193
31	StreamingMovies_No internet service	286.520193
18	OnlineBackup_No	284.074903
21	DeviceProtection_No	251.672514
34	Contract_One year	176.123171
17	OnlineSecurity_Yes	147.295858
26	TechSupport_Yes	135.559783
1	SeniorCitizen	134.351545
3	Dependents	133.036443
6	PaperlessBilling	105.680863
37	PaymentMethod_Credit card (automatic)	99.582057
2	Partner	82.412083
36	PaymentMethod_Bank transfer (automatic)	76.485913
30	StreamingMovies_No	72.898756
9	InternetService_DSL	71.313180
27	StreamingTV_No	70.349516

Figure 5.6: The 30 most relevant features according to Univariate selection.

Recursive Feature Elimination (RFE) is a feature selection approach that works by recursively removing variables and building a model the remaining variables. Accuracy is the metric used to identify the attributes that contribute the most to the more accurate

prediction of the target attribute. In this case, we choose the 20 most effective in accuracy features which are depicted in figure 5.7. It is obvious that there are many common features among them.

	features
0	PaperlessBilling
1	InternetService_DSL
2	InternetService_Fiber optic
3	InternetService_No
4	MultipleLines_No
5	OnlineSecurity_No internet service
6	OnlineBackup_No internet service
7	DeviceProtection_No internet service
8	TechSupport_No internet service
9	StreamingTV_No
10	StreamingTV_No internet service
11	StreamingMovies_No
12	StreamingMovies_No internet service
13	Contract_Month-to-month
14	Contract_Two year
15	PaymentMethod_Credit card (automatic)
16	PaymentMethod_Mailed check
17	tenure
18	MonthlyCharges
19	TotalCharges

Figure 5.7: The 20 features selected via RFE.

The last technique we use to select features is an embedded technique, which is based on the feature importance of XGBoost classifier and SelectFromModel which is a scikit-learn's class that takes a model and is able to transform a dataset into a subset with selected features.

More specifically, we first train and then evaluate the XGBoost model on the entire training dataset and the test dataset respectively. After that, we use these feature importance scores as thresholds, and starting with all features we end up in a subset of the most effective attributes depending on these thresholds. This is an idea from [35].

Chapter 6

Results and Conclusions

In this chapter, we show the results of the models we built, after being trained and tested in the dataset [24]**.** Also, we compare these results with the corresponding ones in surveys [3] and [4]**.**

6.1 Train and test the algorithms with bare data

A first try was to use the data in the form they had after the pre-processing stage. That is, we applied neither the imbalanced techniques nor the feature selection. The results are depicted in figure 6.1, and as it was expected the scores of the used metrics are very low. More specifically, there are six columns, Model, Prediction_Score, Precision, Recall, F1_score, and AUC_score. The Model column includes the names of the learning algorithms and the Precision_Score is the accuracy of the model. Looking carefully, it seems that accuracy's and AUC scores are just satisfying, with the highest values in 80.9% by LOG_REG and 74.7% by NB respectively. The other metrics are quite better than the measures of a random classifier with an exception of the recall score NB succeed, which is 84.77%.

	Model	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	LOG_REG	0.809084	0.697161	0.560914	0.621660	0.733166
1	KNN	0.765082	0.594595	0.502538	0.544704	0.684767
2	SVM	0.800568	0.679365	0.543147	0.603667	0.721820
3	NB	0.702626	0.481962	0.847716	0.614535	0.747011
4	RF	0.776437	0.666667	0.401015	0.500792	0.661591
5	XGB	0.804116	0.697987	0.527919	0.601156	0.719624

Figure 6.1: metrics results after predicting in bare data.

6.2 Apply imbalanced techniques and evaluate predictions

The main reason for these undesirable results in figure 6.1, is that the dataset is imbalanced. That is, the two classes, Churn and Non-Churn customers, are not represented equally, and specifically, the ratio of these two classes' instances is 2.77:1. In addition, the learning algorithms predict almost always that a customer will not churn and the accuracy score is that good. This phenomenon is called accuracy paradox and is the reason give more weight to the other metrics, like precision, recall and AUC scores.

Subsequently, to have a better performance we apply the imbalanced techniques, SMOTE and ADASYN. In figure 6.2 below, there are two tables, the upper one contains the scores applying ADASYN and the lower one the scores applying SMOTE. The format of each table is the same as in figure 6.1. In comparison with the results in figure 6.1, there is a great improvement in both ADASYN and SMOTE methods. As we have already mentioned in chapter 5 ADASYN is an oversampling technique that tries to make new synthetic data points by these points which would be difficult for the classifier to predict. On the opposite way, SMOTE makes the new synthetic data points by using the KNN algorithm and therefore select information from the nearest neighbors. Hence, the predictions' scores via SMOTE method should be better than ADASYN and actually they are, as seen in figure 6.2. The trade-off between these two methods is that SMOTE can achieve higher evaluation results but on the other hand ADASYN creates new samples in a more realistic way – instead of being linearly correlated to the parents they have a little variance in them.

In particular, the highest precision score is performed by RF algorithm in both imbalanced techniques, 85.93% in ADASYN and 86.68% in SMOTE. The algorithm with the best recall score is KNN, with 93.49% in ADASYN and 91.76% in SMOTE and is one of the two cases where applying ADASYN returns better results than applying SMOTE. The other one is the XGB's precision score, which is 83.39% in ADASYN and 82.98% in SMOTE. However, the metric which helps us understand, which algorithm performs better is the f1 score, and in this metric the best performance is coming from XGB classifier, 84.22% in ADASYN and 84.51% in SMOTE, but also RF's f1 score are close enough to these values, 83.21%, and 84.3% corresponding.

ADASYN:

	Model	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	LOG_REG	0.749274	0.730131	0.800000	0.763470	0.748678
1	KNN	0.782672	0.719440	0.934928	0.813150	0.780882
2	SVM	0.719748	0.672849	0.867943	0.758044	0.718007
3	NB	0.710068	0.674491	0.824880	0.742144	0.708718
4	RF	0.835431	0.859327	0.806699	0.832182	0.835768
5	XGB	0.838819	0.833959	0.850718	0.842255	0.838679

SMOTE:

	Model	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	LOG_REG	0.779227	0.756279	0.806548	0.780605	0.779922
1	KNN	0.804348	0.741780	0.917659	0.820399	0.807229
2	SVM	0.746860	0.697712	0.847222	0.765233	0.749411
3	NB	0.745411	0.696648	0.845238	0.763783	0.747949
4	RF	0.851208	0.866876	0.820437	0.843017	0.850425
5	XGB	0.846377	0.829828	0.861111	0.845180	0.846751

Figure 6.2: Metrics results after applying SMOTE and ADASYN technique.

In the sections below, we use the oversampled data that come from the SMOTE technique for the development and the improvement of our model.

6.3 Apply feature selection and evaluate predictions

In the previous section, despite the fact that we had great evaluation results, we perform predictions using all the features of the dataset. This leads to more training and testing time and possibly to non-generalized results. To avoid these issues we apply feature selection techniques on the SMOTE's oversampled data.

First, we try the 30 most relevant features that univariate selection returns, as we saw in chapter 5. The results are depicted in figure 6.3. It seems that the scores are lower compared to the scores in figure 6.2. Meanwhile, XGB classifier performs the best scores in total that are highlighted blue.

	Model	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	LOG_REG	0.775845	0.752788	0.803571	0.777351	0.776550
1	KNN	0.790821	0.757386	0.839286	0.796235	0.792053
2	SVM	0.746860	0.697712	0.847222	0.765233	0.749411
3	NB	0.729952	0.665928	0.893849	0.763236	0.734119
4	RF	0.825604	0.838743	0.794643	0.816098	0.824817
5	XGB	0.828019	0.806391	0.851190	0.828185	0.828608

Figure 6.3: Evaluation metrics after applying univariate feature selection.

Next, we apply RFE's 20 selected features on the same oversampled data and the evaluation metrics depicted in figure 6.4. Once again, the best performance –highlighted blue - is coming from XGB algorithm. That is, it seems to be the most robust and consistent classifier among all others that used in this research. Due to this fact, the following study includes XGB algorithm.

The scores are even lower than scores in figure 6.3. As seen, the more we decrease the number of features the more the metrics get lower scores. In addition, the previous two feature selecting methods require to set the number of desirable dimensions of the new subset manually. But which is the best number of features to select is something that we can not know in advance.

A way to deal with this issue is the embedded feature selection method, that we described in chapter 5. We keep the best learning algorithm so far, the XGB, and using its feature importance to create thresholds.

Generally, importance provides a score that indicates how useful or valuable each feature was in the construction of the boosted decision trees within the model. The more an attribute is used to make key decisions with decisions trees, the higher its relative importance.

In particular, having all the features of the dataset and using an iterative method, reduce them depending on the threshold value. That is, starting with importance score equal to zero means that all features are included for training and prediction. In next iteration, threshold variable takes the next greater importance score and thus exclude features with zero importance.

These thresholds lead us to the best evaluation metrics' values with the lower possible number of features. Figure 6.5 depicts the results that XGB performs in each iteration, starting from the fortieth feature and ending up to only one. It is observed that by keeping the twenty-two most important features, the learning algorithm achieves the best performance in these evaluation metrics. The optimal scores are highlighted blue and particularly are:

- **Accuracy:** 84.97 %
- **Precision:** 83.22 %
- **Recall:** 86.60 %
- **AUC:** 85.01 %

In addition, these twenty - two features are shown in figure 6.6.

	Model	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	LOG_REG	0.772947	0.752345	0.795635	0.773385	0.773524
1	KNN	0.783575	0.751799	0.829365	0.788679	0.784739
2	SVM	0.746860	0.697712	0.847222	0.765233	0.749411
3	NB	0.728502	0.664454	0.893849	0.762267	0.732706
4	RF	0.804348	0.815707	0.772817	0.793683	0.803546
5	XGB	0.819807	0.799246	0.841270	0.819720	0.820352

Figure 6.4: Evaluation metrics after applying RFE feature selection.

6.4 Comparing model performance

In chapter 2, we refer to already published papers that provide developed models to predict customers' churn. Especially, in [3] four different datasets were used, but one of them [24] is the one we also use in this thesis. Therefore, we can compare the performance results with respect to the common metrics in both studies. These metrics are accuracy, recall, precision, and f1 scores and the comparison of them is seen in table 6.1. Although the best performance of the model suggested in [3] is quite better than the best performance of our model, by looking at the f1 scores, the values of metrics are close enough.

Model	Accuracy	Precision	Recall	F1
XGB	**0.849**	0.832	**0.866**	0.848
Gen	0.848	**0.881**	0.794	**0.858**

Table 6.1: Comparison the XGB's evaluation metrics with Gen's ones [3].

Another possible comparison is with the model developed in [4], as A.Amin, F. Al-Obeidat, B.Shah, A.Andan, J.Loo, and S.Arwar also the same dataset. Their implementation is about two zones the upper and lower zones, to estimate the expected certainty of the classifier. The customers in the lower zone show uncertain behavior as compared to upper zone customers. They used as evaluation metrics accuracy, precision, recall, and f1 scores, too. The results are shown in table 3 in [4] and other than the recall metric which includes high scores, especially in the upper zone, the rest of the metrics are significantly lower than our results.

	Prediction_Score	Precision	Recall	F1_score	AUC_score
0	0.846377	0.829828	0.861111	0.845180	0.846751
1	0.846377	0.829828	0.861111	0.845180	0.846751
2	0.846377	0.829828	0.861111	0.845180	0.846751
3	0.846377	0.829828	0.861111	0.845180	0.846751
4	0.846377	0.829828	0.861111	0.845180	0.846751
5	0.846377	0.829828	0.861111	0.845180	0.846751
6	0.846377	0.829828	0.861111	0.845180	0.846751
7	0.846377	0.829828	0.861111	0.845180	0.846751
8	0.846377	0.829828	0.861111	0.845180	0.846751
9	0.846377	0.829828	0.861111	0.845180	0.846751
10	0.846377	0.829828	0.861111	0.845180	0.846751
11	0.846860	0.831256	0.860119	0.845441	0.847197
12	0.843961	0.828380	0.857143	0.842516	0.844296
13	0.843961	0.828380	0.857143	0.842516	0.844296
14	0.846377	0.831731	0.858135	0.844727	0.846676
15	0.845894	0.829035	0.861111	0.844769	0.846281
16	0.845411	0.825758	0.865079	0.844961	0.845911
17	0.846377	0.826705	0.866071	0.845930	0.846878
18	0.849758	0.832221	0.866071	0.848809	0.850173
19	0.848792	0.831268	0.865079	0.847837	0.849206
20	0.849275	0.832696	0.864087	0.848101	0.849652
21	0.846860	0.828735	0.864087	0.846042	0.847298
22	0.845894	0.828408	0.862103	0.844920	0.846306
23	0.846377	0.830460	0.860119	0.845029	0.846726
24	0.844444	0.828544	0.858135	0.843080	0.844793
25	0.843961	0.829011	0.856151	0.842362	0.844271
26	0.843478	0.828215	0.856151	0.841951	0.843800
27	0.840097	0.824545	0.853175	0.838615	0.840429
28	0.817391	0.800573	0.832341	0.816148	0.817771
29	0.816908	0.803279	0.826389	0.814670	0.817149
30	0.812077	0.797310	0.823413	0.810151	0.812365
31	0.811111	0.792972	0.828373	0.810286	0.811550
32	0.810628	0.801961	0.811508	0.806706	0.810650
33	0.807246	0.795344	0.813492	0.804316	0.807405
34	0.794686	0.778947	0.807540	0.792986	0.795013
35	0.780193	0.772952	0.776786	0.774864	0.780107
36	0.780193	0.762583	0.796627	0.779233	0.780611
37	0.763768	0.729038	0.819444	0.771602	0.765184
38	0.757488	0.743738	0.765873	0.754643	0.757701
39	0.730918	0.664000	0.905754	0.766261	0.735363

Figure 6.5: Evaluation metrics applying embedded feature selection method to XGB classifier.

	features
18	DeviceProtection_No
19	DeviceProtection_No internet service
20	DeviceProtection_Yes
21	TechSupport_No
22	TechSupport_No internet service
23	TechSupport_Yes
24	StreamingTV_No
25	StreamingTV_No internet service
26	StreamingTV_Yes
27	StreamingMovies_No
28	StreamingMovies_No internet service
29	StreamingMovies_Yes
30	Contract_Month-to-month
31	Contract_One year
32	Contract_Two year
33	PaymentMethod_Bank transfer (automatic)
34	PaymentMethod_Credit card (automatic)
35	PaymentMethod_Electronic check
36	PaymentMethod_Mailed check
37	tenure
38	MonthlyCharges
39	TotalCharges

Figure 6.6: The 22 most important feature according to XGB's feature importance.

6.5 Conclusion

In this thesis we developed and compared to each other, six different machine learning algorithms, LOG_REG, KNN, SVM, NB, RF, and XGB. In addition, we applied feature selection techniques as well as imbalanced data handling methods, to exploit the data in the most efficient way. The evaluation metrics that we used are Accuracy, Precision, Recall, F1 and AUC scores. After a step by step analysis on the results of these metrics, the XGB classifier outperforms the other used classifiers. We also, provided comparisons between this model and models developed in [3], [4], and we show that it is a competitive model that

is able to predict customers churn decently. As future work, we aspire to improve the metrics we used, by properly tuning the XGB's parameter, which consists a hard process. Moreover, we aim to focus on the attributes of customers and try to infer in a more sophisticated way which of them are crucial and to what extent these features affect the model.

References

[1] R.Mattison, *The Telco Churn Management Handbook*, Oakwood Hills, Illinois, USA, XiT Press, 2005.

[2] D.A. Chiang, Y.F. Wang, S.L. Lee, C.J. Lin, 'Goal-oriented sequential pattern for network banking churn', *Expert Systems with Applications*, 25, (2003) 293-302.

[3] A.Amin, et al, "*Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study*", IEEE Access, vol. 4, p. 7940-7957, Oct. 2016, DOI: 10.1109/ACCESS.2016.2619719.

[4] A.Amin, F.l-Obeidat, B.Shah, A.Adnan, J.Loo, S.Anwar, "*Customer churn prediction in telecommunication industry using data certainty*", Elsevier - journal of business research, vol. 94, p. 290-301, Jan 2019, https://www.sciencedirect.com/science/article/pii/S0148296318301231, (Accessed: 29/6/2019).

[5] N.S.Nafis, M.Makhtar, M.K.Awang, M.N.Rahman, M.M.Deris, "*Predictive modeling for telco customer churn using rough set theory*", ARPN Journals, vol. 11, no. 5, Mar. 2016, https://www.researchgate.net/publication/309579526_Predictive_modeling_for_telco_customer_churn_using_rough_set_theory, (Accessed: 29/6/2019).

[6] KDnuggets, "*The essence of Machine Learning*", https://www.kdnuggets.com/2018/12/essence-machine-learning.html, (Accessed: 30/6/2019).

[7] T.Mitchell, "*Machine Learning*", McGraw Hill, 1997

[8] I.Goodfellow, Y.Bengio, A.Courville, "*Deep Learning*", The MIT Press, November 18, 2016, chapter 5, p.98

[9] D.Conway, Data Science Venn Diagram, http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram (Accessed: 10/6/2019).

[10] J.Brownlee, "*A gentle introduction to k-folds Cross-Validation*", https://machinelearningmastery.com/k-fold-cross-validation/ (Accessed: 24/6/2019).

[11] scikit-learn, "*Cross-validation: evaluating estimator performance*", https://scikit-learn.org/stable/modules/cross_validation.html, (Accessed: 28/6/2019).

[12] Wikipedia, "*Logistic Regression*", (Accessed: 20/6/2019).

[13] S.Sayad , "*K Nearest Neighbors - Classification*", https://www.saedsayad.com/k_nearest_neighbors.htm (Accessed: 24/6/2019)

[14] B.DeWilde, "Classification of Hand-written digits", https://bdewilde.github.io/blog/blogger/2012/10/26/classification-of-hand-written-digits-3/, (Accessed: 1/7/2019).

[15] J.Brownlee, "*Support Vector Machines for Machine Learning*", https://machinelearningmastery.com/support-vector-machines-for-machine-learning/, (Accessed: 24/6/2019)

[16] Figure 7, Maximum margin classification with support vector machines, https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781783555130/3/ch03lvl1sec21/maximum-margin-classification-with-support-vector-machines, (Accessed: 26/6/2019).

[17] Wikipedia, *Naïve Bayes Classifier*, https://en.wikipedia.org/wiki/Naive_Bayes_classifier, (Accessed: 26/6/2019)

[18] V.Smolyakov, *Ensemble Learning to Improve Machine Learning Results*, M, 2017, https://blog.statsbot.co/ensemble-learning-d1dcd548e936, (Accessed: 26/6/2019).

[19] Figure 8, *Ensemble Methods in Machine Learning: What are They and Why use them?*, https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f, (Accessed: 26/6/2019).

[20] A comprehensive guide to Ensemble Learning, https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/, (Accessed: 20/6/2019).

[21]T.Chen, C.Guestrin, "*XGBoost: A Scalable Tree Boosting System*", University of Washington, 2016, https://arxiv.org/pdf/1603.02754.pdf, (Accessed:26/6/2019).

[22] Measuring Performance: AUC(AUCROC), https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/, (Accessed: 26/6/2019).

[23] Kaggle, Wikipedia, https://en.wikipedia.org/wiki/Kaggle,

[24] Telcom Customer Churn, Kaggle, https://www.kaggle.com/blastchar/telco-customer-churn#WA_Fn-UseC_-Telco-Customer-Churn.csv

[25] Python Documentation, https://docs.python.org/3/tutorial/index.html

[26] Anaconda Documentation, https://docs.anaconda.com/

[27] Jupyter Notebook Documentation, https://jupyter-notebook.readthedocs.io/en/stable/notebook.html

[28] R.Vasudev, "*What is One Hot Encoding? Why and when do you have to use it*", Hackernoon, Medium, https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f, (Accessed: 26/6/2019).

[29] sklearn.preprocessing.StandardScaler, https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html, (Accessed: 20/6/2019).

[30] J.Brownlee, "*8 Tactics to combat imbalanced classes in your machine learning dataset*", https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/, (Accessed: 22/6/2019).

[31] N.V. Chawla, K.W.Bowyer, L.O. Hall, W.P. Kegelmeyer, "*SMOTE: Synthetic Minority Over-sampling Technique*", Journal of Artificial Intelligence Research, 2002, v.: 16, p.: 321–357, https://www.jair.org/index.php/jair/article/view/10302/24590, (Accessed: 24/6/2019).

[32] S.Paul, "*Diving Deep with Imbalanced Data*", DataCamp, 2018, https://www.datacamp.com/community/tutorials/diving-deep-imbalanced-data, (Accessed: 20/6/2019).

[33] J.Brennan, "*Dealing with Imbalanced Data*", Newcastle University, 2018, https://www.migarage.ai/intelligence/imbalanced-data/, (Accessed: 25/6/2019).

[34] Wikipedia, "*Feature Selection*", https://en.wikipedia.org/wiki/Feature_selection, (Accessed: 30/6/2019).

[35] J.Brownlee, "*Feature Importance and Feature Selection with XGBoost in python*", https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/, (Accessed: 30/6/2019).