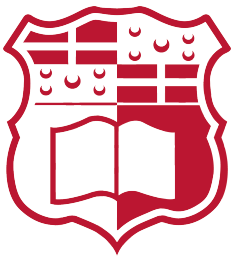
Investigating Training Set and Learning Model Selection for Churn Prediction in Online Gaming

LUKE BORG

Supervised by Dr Jean-Paul Ebejer
Co-supervised by Dr John Abela



Faculty of ICT
University of Malta

July, 2021

Submitted in partial fulfilment of the requirements for the degree of Master of Science in Artificial Intelligence

L-Università
ta' Malta

WWW.UM.EDU.MT

To my soon to be wife, Denise

For being my constant in every step of the way.

Acknowledgements

Throughout this research, I have received support and assistance from several persons to whom I would like to express my gratitude. First and foremost, I would like to express my most profound appreciation to my supervisor Dr Jean-Paul Ebejer and my co-supervisor Dr John Abela for providing guidance, feedback and a wealth of knowledge which steered me in the right direction. I would also like to thank the industrial collaborator for providing invaluable help and discussions on the churn prediction domain that made this work possible.

Also, I would like to thank my parents for encouragement during challenging times. I could not have finished this dissertation without the constant support of my better half Denise. Thank you for always being there.

Abstract

Customer retention has always been a key performance indicator for businesses, as it has a direct impact on profit and revenue. In order for a business to control customer retention, the customer churn rate needs to be controlled. Customer *churn*, is the action of a customer dissociating itself from the business, hence directly affecting customer retention. Machine Learning algorithms are gaining momentum in order to control customer retention and churn. Furthermore, data related to customer churn possesses changing trends and variability (denoted by *concept drift*), which decreases the machine learning model predictive power. This decrease in predictive power is noted when predicting distant periods from the learning period and is formally known as the model's staying power.

This research investigates the effect of concept drift on the mobile gaming and iGaming industries. A benchmark paper which uses data from the mobile gaming industry is used to investigate concept drift. Additionally, the investigation of concept drift in the iGaming industry is carried out through an industrial collaboration. Three concept drift mitigation approaches are used to address the problem of concept drift being the *Moving Window* approach, the *Incremental Window* approach and the *Window Dissimilarity* approach. Several machine learning models are used, namely Random Forest (RF), XGBoost and Light Gradient Boosting Machine (LGBM) (amongst others). Results show that in the case of the benchmark paper, these approaches did not have any impact on performance. However, in the case of the industrial collaborator, a statistically significant improvement is noted as the RF obtains a ROC-AUC 0.741, the LGBM 0.755 and XGBoost 0.752 for the moving window approach while for the window dissimilarity approach the RF achieves 0.740, LGBM 0.754 and XGBoost 0.752. An improvement in ROC-AUC of 3% is noted when considering the results of the industrial collaborator model.

Contents

List of Figures

List of Tables

List of Abbreviations

1

Introduction

A stable and loyal customer base helps companies and businesses prosper. However, in this technological era, retaining a loyal customer base proves to be a difficult task mainly due to competition. For this reason, companies embrace the use of Customer Relationship Management (CRM) processes in order to place their customers as their central focal point. Customer retention is the fundamental focus, and purpose, of CRM, because this is a measure of the customer's satisfaction. A CRM system allows a comparison between the customer's expectations and the customer's perception of satisfaction. The former relates to what the customer expects from the service offered while the latter refers to if those expectations were satisfied after the customer thoroughly used the service (Chau et al., 2009; Teal, 2001).

1.1 Motivation

One of the essential metrics for determining the customer retention capabilities of a business is *churn*. Customer churn signifies the action of a customer's disassociation with a company or business (Chu et al., 2007). The issue of customer churn is encountered in different industries and businesses - be it financial, retail, telecommunication, gaming, and any business that has direct contact with the customer. The World Wide Web (WWW) has empowered the customers since it is now straightforward, and effortless, for a customer to switch from

one vendor to another. On the other hand, companies generally store all the customer data, be it related to transactional activity, purchases, and so on. This customer data is a gold mine which can be used in designing effective customer retention strategies. In fact, in this era of big data, emergent technologies are being used to elicit pertinent customer information from the large volumes of data. In particular, big data technologies are used in the banking industry to identify patterns which lead to customer churn from massive noisy data. Furthermore, a significant amount of research has focused on the problem of churn prediction, and this has led to the adaption of machine learning methodologies to improve customer retention (Cui and Ding, 2017; Do et al., 2017).

The definition of customer churn varies across different business domains. Furthermore, it depends on many factors such as the type of product used, if the service is subscription-based or not, amongst others. Generally, in cases of subscription-based services, a customer is defined as churned when the subscription is cancelled (Do et al., 2017). However, when it comes to the retail industry, the definition of churn is defined as a period where the customer does not use the service offered. This period needs to be carefully chosen as this will have severe repercussions in the outcome of the retention strategy. In the case of the retail industry, a typical retention strategy would be to give vouchers to stagnant customers. Albeit, this is done after carefully evaluating the appropriate period to do so, as this could lead to the repercussion of wasting valuable funds (Deepshika et al., 2017). In the banking world, the typical definition of churn is when the bank customer withdraws all the money and closes down the accounts (Cui and Ding, 2017). Furthermore, different domains carry different churn ratios. Literature shows that in the telecommunications industry, the churn rate typically hovers around 30% to 40% of the total customers considered (Cankaya et al., 2012).

The problem of customer churn is not one that should be taken lightly because it has severe implications on the business's revenue and growth. Having a high churn rate implies channelling more funds for customer acquisition, and studies show that acquiring new customers amounts to almost five times higher than retaining existing ones (Liu and Ng, 2000). Hence, this is the driving force behind the research available on using predictive analytics and machine learn-

ing in churn prediction. However, in general, research about customer churn does not consider the staying power of the model (Ahmed and Linen, 2017). Staying power refers to how the machine learning model performs in future periods that are far away from the training period and defines how the predictive performance of a machine learning model varies when predicting on periods far off from the training period. Studies show that machine learning models experience loss of staying power which indicates that the predictive performance degenerates when testing on distant periods. According to research, such comportment is tightly linked with the underlying nature of the data (Bijmolt et al., 2010). Generally, data used for churn prediction varies continuously. This variation leads to emerging concepts in the data, which is known as concept drift. The term concept drift is defined as unanticipated changes in the underlying structure of the data as time changes. In other words, patterns or variations present in past data may not be pertinent to incoming data, thus leading to performance issues. Thus, concept drift leads to degradation of the model's predictive performance, ultimately affecting the model's staying power (Liu et al., 2018).

The focus of this study is two-fold. The first being to analyse the effect of variation in the data present in mobile games data. In order to do so, the benchmark paper of Choi et al. (2017) was selected, with the choice based on firstly, dataset availability. It is unusual to find such datasets online. Secondly, the dataset exhibited severe class imbalance, and lastly, the problem of data variability was not addressed in the paper. Another reason for selecting this paper was our intention to analyse the cross-domain compatibility of the data variability (concept drift) approaches that are applied to the mobile games dataset. By cross-domain, we mean whether an approach that works on one domain (example mobile games industry), works equally well on another domain such as the iGaming industry. In the second part of this work, we collaborated with an industry partner from the iGaming industry that allowed access to its real-life customer data which was used to evaluate these approaches.

1.2 | Aims and Objectives

The main aim of this research is to create a churn prediction system based on machine learning that can deal with data variation and concept drift in the data. The objectives of this study are:

1. Analyse the dataset to identify patterns related to concepts and data variability both in the work of Choi et al. (2017) and data provided by the industrial collaborator.

2. Implement concept-drift detection through the use of windowing techniques. For each technique, apply different machine learning models and evaluate vis-à-vis the results obtained by Choi et al. (2017) and the industrial collaborator.

3. Evaluate cross-domain compatibility between the mobile gaming and iGaming industries, with the latter involving an industrial collaborator.

1.3 | Approach

The first part of this dissertation builds on the work of Choi et al. (2017), which focuses on predicting customer churn in the mobile games industry. The mobile games industry recently has seen some radical improvements in game development tools resulting in increased competition between game developers, hence the need for churn prediction. According to Choi et al. (2017), the application of churn prediction algorithms to such data has been relatively limited in past research, which is attributed to the fact that the mobile game industry is an emergent one. The originality of this dissertation that contrasts the work of Choi et al. (2017) is situated in the data pre-processing part. In this dissertation, the addition of a date field known as the decision date permits dataset ordering. As the name implies the decision date is that particular date on which a churn prediction is taken. Based on this prediction, a player is deemed as a churner or not. This decision date is used first to identify the phenomenon of concept drift and

secondly to implement the concept drift aware approaches. To determine concept drift, two experiments are carried out. The first makes use of a Augmented Dickey Fuller (ADF) statistical test which gauges if a time-series is stationary or not. Non-stationarity indicates the evidence for changing trends and variability (concept drift) in the data. The second makes use of a decision tree complexity to measure the spread of this data variability. During this experiment, decision tree properties such as the node count and depth are analysed. After concept drift identification, two concept drift mitigation approaches are used. The first being the Moving Window approach while the second is termed as the Incremental Window approach. In the first approach, a window of a specific size be it in days or rows is moved along the evaluation period. For each window, the machine learning model predicts the future period, which is also of a specific length. Contrastingly, in the Incremental Window approach, the model starts with a predefined window size as in the case of the first approach. However, as the window is moved across the evaluation period, the data from past windows is also used for model training hence the term incremental. Thus, in the second approach, the model uses more data for training purposes after the first window.

The second part of this dissertation involves an industrial collaboration with a company from the iGaming industry. Similar to the first part of this study, the dataset is enhanced to permit dataset ordering. In this case, the First Deposit Date (FDD) is used to order the dataset. The first phase of this part revolves around experiments that elicit trends and variability in the data. This is carried out using statistical tests such as the ADF and Kwiatkowski Phillips Schmidt Shin (KPSS) in order to determine if the time series is stationary or not. In addition to this test, another two experiments are used to investigate concept drift, which are carried out using a decision tree. In the first decision tree experiment, the tree properties (node count and depth) are analysed in relation to other dataset properties such as the number of customers present and how many of them churn. In the second phase, two concept drift mitigation approaches are implemented. The first is the Moving Window approach which is conceptually similar to the one mentioned in the first part of this dissertation. In contrast, the second approach is termed as the Window Dissimilarity Approach. The window dissimilarity approach adopts the same moving window technique as the

first approach, however, only if concept drift is detected, the machine learning model is updated. The data variability or drift is detected through the use of the Kolmogorov-Smirnov (KS) statistical test. During the evaluation, the results from these two approaches are compared against the churn prediction model of the industrial collaborator. Furthermore, by applying these different concept drift aware techniques on two different domains, the cross-domain compatibility (that is whether such approaches apply to both domains) is discussed.

1.4 | Contributions

The main contributions of this dissertation are:

1. An analysis of data variability and concept drift in the context of mobile gaming and iGaming industries.
2. Three different types of concept-drift mitigation approaches – Moving Window, Incremental Window and Window Dissimilarity.
3. The application of the above approaches to two different domains to investigate the effect of concept drift.

1.5 | Document Structure

This document is structured as listed below. Readers who are familiar with customer churn prediction and machine learning may choose to skip Chapter 2.

Background (Chapter 2) The background section will offer the reader all the relevant research required for understanding future sections. It gives a detailed discussion on churn prediction and concept drift handling mechanisms. Furthermore, it reviews various machine learning algorithms in detail as applied to churn prediction.

Literature Review (Chapter 3) The review chapter will offer the reader the main work and contributions in relation to customer churn prediction. This chapter

also gives an in depth review on the research paper of Choi et al. (2017) which is used as the benchmark paper.

Methodology (Chapter 4) The focus of this work then shifts to the iGaming partner. Firstly experiments and concept drift approaches are applied on the work of Choi et al. (2017) based on the concluding remarks of Chapter 3. Then a similar approach is used to analyse the industrial partner dataset, where first it analyses if the dataset contains concept drift. Secondly, it discusses two approaches to address the issue of concept drift. The first approach is the moving window approach which was also used during the investigation of the work of Choi et al. (2017) and the second approach is the window dissimilarity approach. It commences by describing in detail the dataset that will be used. Moving on to the pre-processing and feature engineering part and finally describing the implementation of the machine learning pipeline. This chapter concludes by detailing the experiments to perform the data analysis and concept drift detection approaches.

Results and Evaluation (Chapter 5) This section, presents and discusses the results for each experiment mentioned in the previous section. Furthermore, it discusses the evaluation of our results with that of the current churn prediction model in operation at the industrial collaborator.

Conclusion (Chapter 6) The final chapter gives a concluding overview of this dissertation, briefly discusses how the problem of concept drift was solved. It also discusses the achieved aims and objectives and highlights any limitations. Furthermore, any potential future work that can be carried out is listed.

2

Background

The purpose of this chapter is to give a thorough introduction to the domain of churn prediction and explains where churn prediction stems. Furthermore, the different machine learning techniques and performance metrics that will be used in later sections of this dissertation will be explained.

2.1 | Customer Relationship Management

Customer churn analysis and prediction fall under the umbrella of Customer Relationship Management (CRM). In literature, there are several definitions of CRM. However, one of the definitions that generalises all aspects of CRM is the one given by Buttle (2004) and reads as follows:

"CRM is the process of managing all aspects of interaction a company has with its customers, including prospecting, sales and service. CRM applications attempt to provide insight into and improve the company/customer relationship by combining all these views of customer interaction into one picture."

There are multiple points of view for a CRM system, such as the analytical and operational view. The former deals with customer analysis and studies the behavioural patterns of the customers, while the latter deals with automating the business processes that have to do with the CRM system (Chau et al., 2009). Throughout the years, technological developments and innovations

sparked major improvements in CRM, such that CRM became a staple part of the companies. In fact, CRM is considered a mantra in the marketing world (Winer, 2001). This led to the subdivision of CRM into four main niches that all revolved around the customer being (Chau et al., 2009; Kracklauer et al., 2004):

1. Identification
2. Attraction
3. Retention
4. Development

The collaborative aim of these four points mentioned above is one, to maximise the customer values for the company in the long run through the thorough study of the customer. Chau et al. (2009) proposes a depiction that groups these subdivisions to close the CRM architecture. This is shown in Figure 2.1. The arrows indicate the customer's journey beginning from customer identification. This phase is termed the customer acquisition phase and deals with targeting new customers from the population which are deemed to become the most profitable for the company. This phase goes beyond trying to acquire a new customer as it involves deep analysis of competing companies who might loot customers. After such an analysis is done, new ways on how to retrieve the lost customer are proposed (Chau et al., 2009; Ling and Yen, 2001). As Figure 2.1 shows, this step consists of target customer analysis and segmentation. In target customer analysis, the most profitable segments for the company are studied, after which the company's customer base is filtered using the new segments. Thus customers with similar characteristics are binned into the same segment.

After the customer identification phase, the phase of customer attraction or acquisition follows. As the name implies, this phase aims to attract new customers. This is done by using the segment information gathered from the previous phase in the company's direct marketing strategy. There are several ways a customer can be contacted in a direct marketing strategy in particular direct mail, which is tailored according to the customer's segment (Chau et al., 2009).

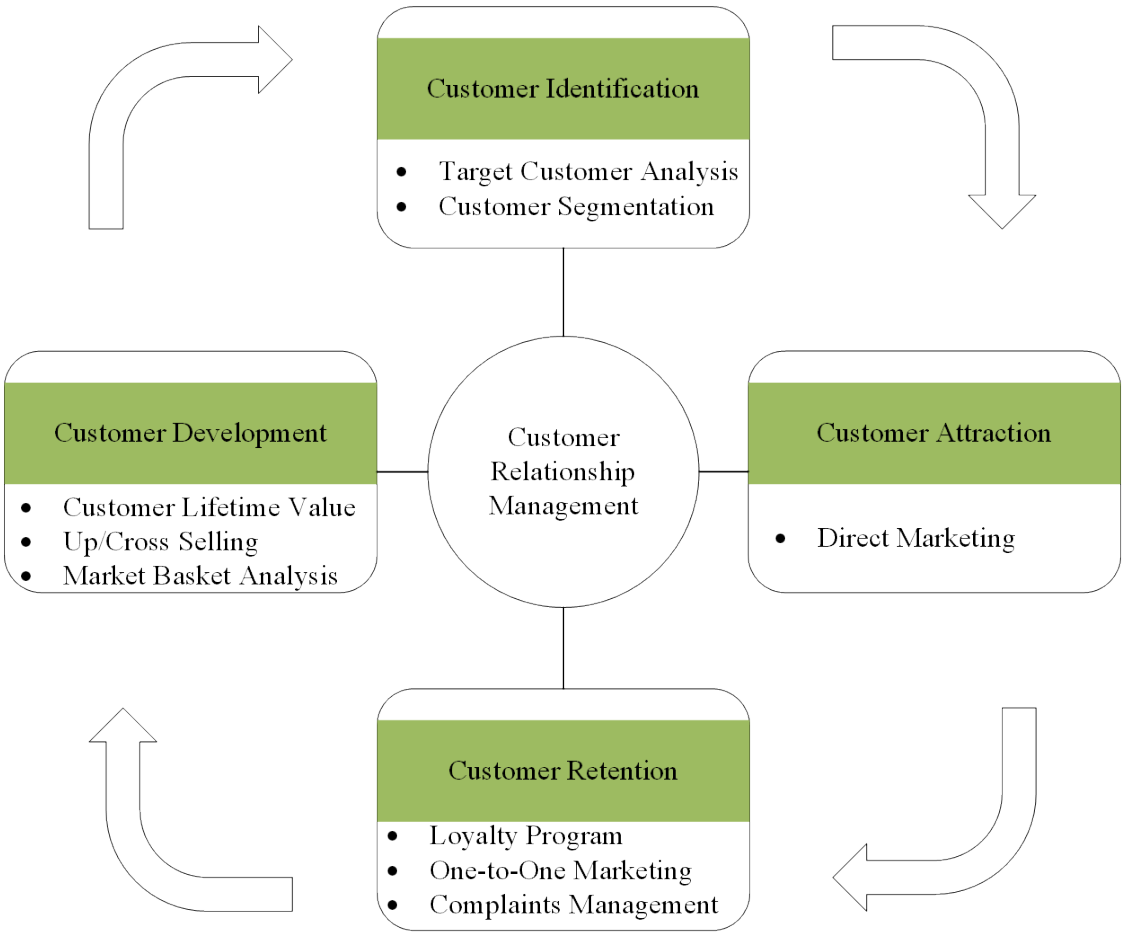


Figure 2.1: CRM Architecture Overview. Adapted from Chau et al. (2009).

The customer retention phase is the next phase after having acquired new customers and segmented them appropriately. As already mentioned earlier in this dissertation, this is a crucial phase of the CRM life cycle shown in Figure 2.1. This phase is challenging and consists of gauging the service quality, churn analysis and types of customer satisfaction programs. Customer behaviour is analysed thoroughly, and customer pattern changes are predicted. The results are then used in marketing campaigns (Chang et al., 2005; Kracklauer et al., 2004).

The final phase to close the CRM life cycle is customer development. This phase also involves customer analysis; however, from a different perspective. In the sense that, in this phase, the customer's past transaction history is analysed to come up with lifetime attributes such as Customer Lifetime Value (CLV).

These lifetime values show the customer's development with time, and such information can then be used to perform market basket analysis (Chau et al., 2009).

2.2 | Customer Churn and Retention

Customer churn, which is sometimes referred to as customer attrition, is defined as the detachment of customer from a product or service offered by a company or a service provider. Thus customer churn is a perfect reflection of customer retention, which is present in the CRM framework. A company or service provider shall aim to have a low churn rate and thus retain the majority of their customer base. Unsatisfied customers will result in a company not able to retain them in the long run, which will contribute to profit loss. Several studies ascertain that companies having high customer retention contribute to higher profits. In contrast, other studies show that gaining a new customer is between five and 25 times more expensive than retaining an existing one (Alamsyah and Salma, 2018; Gallo, 2014; Hou et al., 2011; Ye and Zhai, 2009).

In order to effectively reduce the customers abandoning the service or product, a thorough study to determine strongly correlating aspects of the data with churn is needed. Such study shall begin from first defining churn as this definition varies according to the service offered, company or domain. For example, in internet service providers, service termination is the clear cut definition of churn (Do et al., 2017). However, there are other scenarios where the definition of churn varies, as is the case for this research. In a mobile game application, the only data that is available is play log data, which reflects the player activity. Hence in this study, this activity will be used to produce the churn label for each player. An important thing to note is that in the research carried out by Choi et al. (2017), the dataset used only contains device id; however, according to them, this represents a unique player. Thus in this work, the words player and device can be used interchangeably. In the case of the industrial collaborator, churn is defined differently when compared to the definition given in the work of Choi et al. (2017); as a concept, it still means a customer which defects, however, the period at which this happens varies greatly. Another critical as-

pect of defining the churn criteria is to determine the amount of time to wait in order to declare a player as churned or not. This interval also depends on the churn application and should be selected appropriately. As an example, in the case of casual mobile games, a player is defined as churned if a predefined time has elapsed from the first play and the player did not play the game. This had to be the case because one cannot wait for an indeterminate amount of time to check if the player has churned or not. One needs to set a compromise between waiting too long or too short. However, in the case of casual mobile games, this seems to be a reasonable compromise because players that do not return after for example ten days would not return until the end of data collection period which is much longer than the ten days.

2.3 | Time Series Analysis Related to Churn

There are many categories and types of data, such as sequential data. Sequential data is a type of data that has been sorted in a specific way. In the case of play log or customer transactional log, the data has been sorted by time, which classifies such data as a time series sequential data (Keyvanpour and Molaei, 2015). Given a time series $T = (t_0, t_1, ..., t_n)$, which represents some dataset for a particular user, having data points which are equally spaced. Consider the scenario when dividing this time series dataset into two adjacent but not overlapping time windows. Let us say that these two windows are labelled the *observatory period* and the *predictory period*, respectively. The observatory period is used for discerning the player behaviour, and in this period, several features are computed. Let us assume that the player p has an N-dimensional vector $\boldsymbol{f}_{pt}$ computed for a specific time period. There are several ways of how to compute the user feature vector such as by considering the user's first play (or first deposit date) or by using a current time window (activity window). These windows are used to filter out the most relevant candidates for churn prediction. Furthermore there needs to be a limit on how long to wait which needs to be carefully chosen based on the application as it might be the case that users that have a long history are included when in fact they need to be considered as churned (Choi et al., 2017). Figure 2.2

underpins the discussion mentioned earlier by depicting the periods mentioned. From the above discussion, two pertinent definitions in relation with this study can be derived:

- Each player in case of a mobile game application or a customer in case of online iGaming has an activity log which is a stream of the actions carried during gameplay or online betting. The activity log is closed once the player or customer stops playing and starts being replenished with data once the user returns.
- In order to consider a player or customer as an active one, the respective activity log needs to have one or more entries for the period under study. If this is not the case, then such player or customer is considered as inactive. Thus in this study, the classification task will be a binary one.

Figure 2.2: Timeline explaining different dataset periods.

2.4 | Machine Learning

The simplest definition one could give to machine learning systems is that they assimilate knowledge from data. According to Domingos (2012), machine learning is backed by the scientific research areas of getting computer technology to think, learn and act as humans would do. This enables machine learning systems to improve autonomously over time by learning and to generalise from

examples. Machine learning systems have permeated in different research areas such as churn prediction as is the case of this study, stock prediction, drug discovery and drug design amongst others. Machine learning is based on the mathematical and statistical theory that goes back in the 18^{th} century, such as Bayes Theorem, least-squares data fitting and Markov chains (BBC, 2019). Machine learning systems can be categorised into two main streams being the Supervised and Unsupervised methods.

2.4.1 | Supervised Machine Learning

The machine learning tasks in this research are of the supervised learning type. In supervised learning, the instances used for prediction are labelled meaning that for every feature set, a predictive output exists. The machine learning algorithm is used to map and learn the relationship between the input feature set and the output. It is supervised because the correct predictions are known a priori during the training phase, and thus the algorithm makes the necessary corrections until the desired performance is achieved.

Supervised learning algorithms can be split into two groups being classification and regression. Classification problems are those whose output variable is categorical such as in the case for this study, churn or non-churn or in the case of email filtering spam or not spam. The cases mentioned are binary classification problems; however, there exist multiclass classification problems when more than two predictive classes are present. On the other hand, regression problems deal with a scenario where the output variable is real and continuous, for example, when predicting the height of a group of students (Brownlee, 2016; James et al., 2017).

The objective of any supervised machine learning system is to create the best possible mapping function between the input feature set and the target variable, in order to generalise for future data. In reality, there is no such perfect supervised learning algorithm, and there will be some predictive error which can be determined using the following equation (Brownlee, 2016):

$$E = Var(\hat{y}) + Bias(\hat{y}) + Var(\epsilon) \tag{2.1}$$

where E is the predictive error, $Var(\hat{y})$ is the variance introduced by the mapping function, $Bias(\hat{y})$ is the bias introduced by the mapping function and $Var(\epsilon)$ is referred to as the irreducible error. From Equation 2.1, it can be deduced that the error is comprised of three constituents being the *Variance*, *Bias* and *Irreducible* errors respectively. Variance refers to the degree of change in the mapping function given a different training dataset. Ideally, in reality, when given new training samples, the mapping function should not vary. Bias refers to the scenario where an inflexible (simple) model approximates a rather complicated problem. Thus bias occurs when one is trying to simplify the mapping function as much as possible and thus uses more assumptions. Finally, as the name implies the irreducible error cannot be reduced and results from unknown factors and uncertainty of a system. These definitions lead to a discussion about what is known as the bias-variance tradeoff (James et al., 2017).

2.4.1.1 | Bias Variance Tradeoff

The lower the values for bias and variance, the better that supervised machine learning is considered to be. However, in reality, this cannot be the case, as these two values are competing, hence the nomenclature of bias-variance tradeoff. Supervised machine learning model tends to follow a trend where the parametric or linear models offer low variance, but high bias. In contrast, non-parametric or non-linear models offer low bias but high variance. Inherently if the bias increases (that is model complexity decreases) the variance will follow a downward trend, the opposite happens if the variance is increased (James et al., 2017).

Figure 2.3 illustrates the tension between the bias and variance curves and also shows the combination of the two curves. One can look at bias from the aspect of unmodelled data hence implies underfitting. On the other hand, the variance can be seen as learning from the noise present in the dataset, which hints to overfitting. The optimal model is the one where the two curves meet, which also happens to be the model with the lowest error.

During the modelling phase, one needs to keep in mind the bias-variance tradeoff specifically how these relate when it comes to testing. In particular, we would like to have enough training data to minimise bias and enough data for

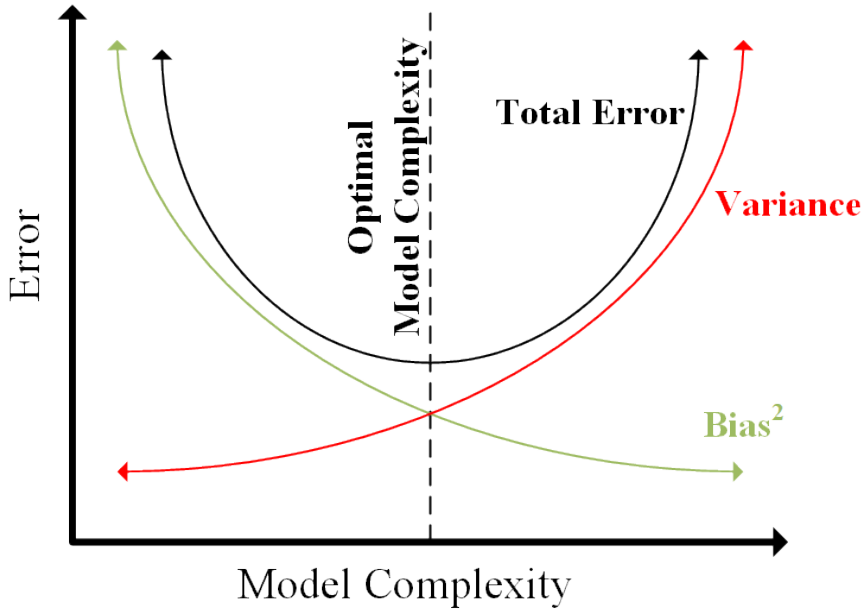


Figure 2.3: Bias-variance tradeoff depiction. Reproduced from Baratin et al. (2018).

testing in order to be able to trust the result and not worry about variance. One way to do this is by using *k-fold cross-validation*. Figure 2.4 depicts an example of five fold cross-validation. This involves a procedure where the dataset is split randomly into k partitions. One of the partitions is chosen for testing, which is also known as the holdout set. The model is fitted on the remaining partitions and then tested on the previously selected holdout partition. The process is repeated k times denoted as iterations in Figure 2.4. Effectively, the result

Iteration 1	Test	Train	Train	Train	Train
Iteration 2	Train	Test	Train	Train	Train
Iteration 3	Train	Train	Test	Train	Train
Iteration 4	Train	Train	Train	Test	Train
Iteration 5	Train	Train	Train	Train	Test

Figure 2.4: Five Fold Cross Validation. Adapted from Fontaine (2018).

would be k performance estimates. Ideally, the model should give low error

rates and thus low bias, and the result shall be consistent, implying low variance. In practice, usually, ten folds are considered, but the only difference is that one will have twice as many folds, the procedure remains the same (Fontaine, 2018; James et al., 2017).

2.4.2 | Unsupervised Machine Learning

In contrast to supervised learning, the unsupervised machine learning variant only makes use of the input feature set without the label or output variable. Thus the aim is to model the underlying data distribution and structure and thus learn about the data. There is no correct way on how to label the instances, and this is left at the algorithm's discretion.

There are two types of unsupervised learning techniques being clustering and association. As the name implies in clustering, the dataset is grouped according to instance similarity. One example would be customer segmentation, where customers are grouped according to their expenditure. In the association type of unsupervised machine learning, the problem in hand is treated as a rule learning problem, where rules are discovered that outline portions of the dataset. A typical example would be that of collaborative filtering where customers are recommended products that were bought by other customers having the same similarity index (Brownlee, 2016; James et al., 2017).

2.5 | Addressing Churn Prediction

This section will give an overview of the different aspects of producing a churn prediction system. These aspects are tightly coupled to the data structure and format, as discussed in the previous section.

2.5.1 | Class Imbalance

As discussed in Section 2.3, the result for such analysis will be to have each row in the dataset with a label; either churn or non-churn. Each row in the dataset

refers to a player or a customer depending on the application. This means that we will have a number of rows labelled as churned and another number of rows labelled as non churned. In practice and as is the case for every real-world scenario, the numbers for each label will not tally. If the labelled data is lopsided, the churn prediction model might not perform as expected. Thus care needs to be taken such that predictions are correctly given. For example, if one class is sparse, the model would not have the predictive power on new data as it is not able to generalise. In the research area of machine learning, the situation when the prediction classes are not almost equal to each other is referred to as class imbalance. One has to note that the severity of class imbalance varies depending on the problem in hand. In order to solve the problem of class imbalance, sampling techniques shall be employed. There is a multitude of sampling techniques, but these can be summarised and grouped into two, being undersampling and oversampling.

In undersampling, the majority class or the class that is the most occurring is downsampled, which in turn leads to achieving a balanced dataset. However, in this case, the dataset size is reduced in size depending on the amount of imbalance present, thus might induce model performance degradation. There are several options for undersampling such as the random undersampling of the majority class, the NearMiss technique and the Condensed Nearest Neighbour Rule amongst others. As the name implies, in the random undersampling strategy, the majority case is undersampled randomly and uniformly. The NearMiss technique was developed to mitigate the problem of information loss. In this technique, the distance between instances of majority and minority class is computed, and those having the smallest distance are selected. The condensed nearest neighbour rule further improves on the aforementioned technique with particular focus on removing instances from the majority class that should not have an impact on model performance. This is carried out by starting from two empty datasets (X and Y). A sample is placed in X while the rest placed in Y. An instance from Y is classified by using X as the training set. If Y was misclassified, it is moved back from dataset Y to X. The process is repeated until no instance transfer happens from Y to X (Adnan et al., 2016; Azhar and Hanif, 2017; Burez and Van den Poel, 2009).

The opposite occurs in oversampling, where the minority class is randomly replicated to produce more instances to equal the majority class. A popular method to mitigate the problem of class imbalance is Synthetic Minority Oversampling Technique (SMOTE). In SMOTE, the minority class is oversampled by creating synthetic instances; while in the generic oversampling technique, pure replication is used to balance the classes. SMOTE works by injecting information from the neighbouring instances to create the new instance, and it is touted to improve generalisation (Adnan et al., 2016; Azhar and Hanif, 2017; Burez and Van den Poel, 2009).

2.5.2 | Concept Drift and Data Variability

In a churn prediction scenario, the idea of having a performing model needs to be reevaluated because of the time element present in the dataset. Consider that for a particular period of the dataset, a model was trained, and results showed that the model achieved a good performance. However, how can one be sure of the model's performance a year or two in the future? This dilemma is known as the model's staying power, and studies show that churn prediction models exhibit a low staying power rendering the model as a short term solution. The modest staying power is attributed to *Concept Drift*, which is tightly coupled to the erratic behaviour of the underlying dataset (Bijmolt et al., 2010).

According to Liu et al., the term concept drift is described as unpredictable variation in the underlying distribution of a time series data. Research shows that when working with streaming data, having varying attributes such as new products launching, new customer behaviours emerging and new market developing leads to concept drift. The reasoning that the model's staying power decreases under concept drift is quite intuitive because the model is presented with totally varying new data from what it was designed for in the first place (Liu et al., 2018).

The formal definition for concept drift is as follows: For a time range $[0, t]$, having a labelled set, denoted as $L_{0,t} = \{l_0, ..., l_t\}$ where $l_i = (\mathbf{X}_i, y_i)$ is a labelled instance composed of a feature vector $\mathbf{X}_i$, y_i as the label and $U_{0,t}(\mathbf{X}, y)$ as the distribution for $L_{0,t}$. A concept is said to occur at a timestamp $t + 1$, if

$U_{0,t}(\mathbf{X}, y) \neq U_{t+1,\infty}(\mathbf{X}, y)$ which can be written as:

$$\exists t : P_t(\mathbf{X}, y) \neq P_{t+1}(\mathbf{X}, y) \tag{2.2}$$

where $P_t(\mathbf{X}, y)$ and $P_{t+1}(\mathbf{X}, y)$ are the probability distributions for the combination of the input feature vector and the labels at time instance t and $t + 1$ respectively.

From Equation 2.2 one can deduce that the joint probability of $P_t(\mathbf{X}, y)$ can be decomposed into two such that $P_t(\mathbf{X}, y) = P_t(\mathbf{X}) \times P_t(y|\mathbf{X})$. This leads to the definition of three types of concept drift as follows (Liu et al., 2018):

- Type 1: $P_t(\mathbf{X}) \neq P_{t+1}(\mathbf{X})$ having $P_t(y|\mathbf{X}) = P_{t+1}(y|\mathbf{X})$. In this case, the probability distribution of the input feature vector changes without any repercussion on the target distribution. This is referred to as virtual drift (Bifet et al., 2014; Liu et al., 2018).

- Type 2: $P_t(y|\mathbf{X}) \neq P_{t+1}(y|\mathbf{X})$ while $P_t(\mathbf{X}) = P_{t+1}(\mathbf{X})$. In this case, the input feature vector distribution does not change, but the target distribution does change. This reflects a decision boundary change and will result in the model to lose accuracy. This is referred to as actual or real drift (Bifet et al., 2014; Liu et al., 2018).

- Type 3: This is a mixture of Type 1 and Type 2 having both $P_t(\mathbf{X}) \neq P_{t+1}(\mathbf{X})$ and $P_t(y|\mathbf{X}) \neq P_{t+1}(y|\mathbf{X})$. This type of concept drift is harder to work with since both probability distribution are changing (Bifet et al., 2014; Liu et al., 2018).

Using Figure 2.5, one could vividly visualise the changing data distribution pertaining to the different types of concept drift. The different ways in which the underlying data distribution of a stream of data can vary is four, being Sudden, Gradual, Incremental and Recurring as illustrated. Additionally, Figure 2.6 highlights the types of concept drift previously discussed.

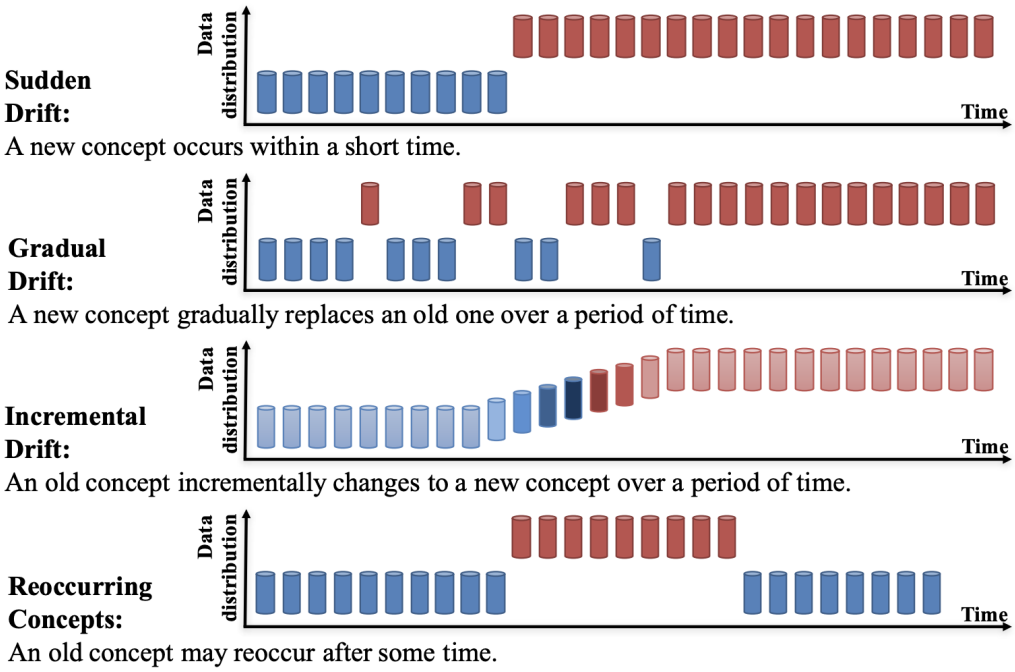


Figure 2.5: Different types of concept drift. Reproduced from Liu et al. (2018).

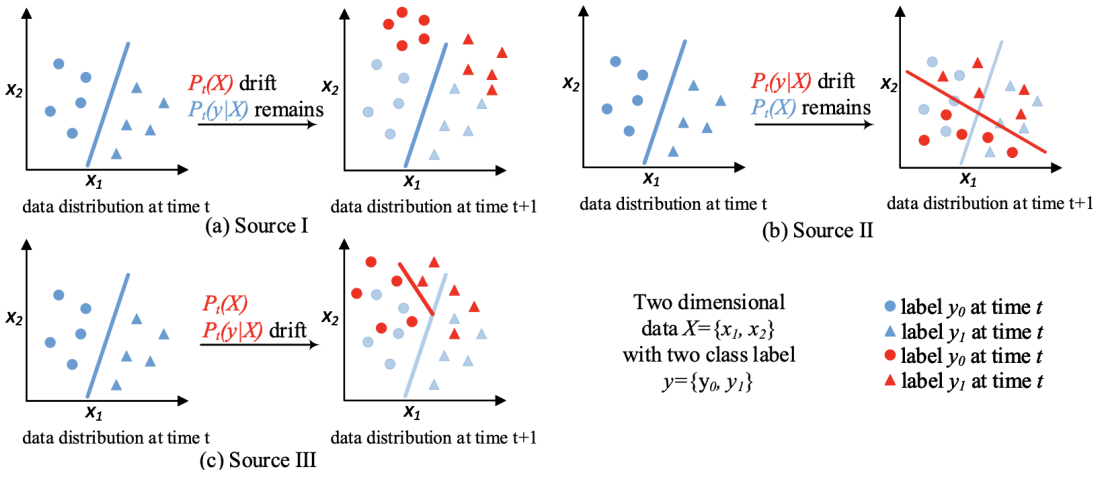


Figure 2.6: Underlying data distribution change for different types of concept drift. Reproduced from Liu et al. (2018).

2.5.3 | Concept Drift Detection Mechanisms

In view of Section 2.5.2, it is evident the need to have adaptive machine learning models. There are several ways in which concept drift can be tackled mainly by using detection methods to measure error rate, by using data distribution drift measures and by using the statistical hypothesis drift detection methods.

When using the error rated drift detection method, the concept drift algorithm tracks the error changes, and if this error rate overshoots a predefined threshold, the model is modified accordingly. In this case, when new data arrives, the algorithm continually checks the error and retrains the models with all the available data if the error is out of bounds (Liu et al., 2018).

When detecting concept drift through the variation of the data distribution, a statistical test is used to measure if there is a statistically significant change between the old data distribution and the new data distribution. If there is a statistically significant difference, a model update is triggered (Liu et al., 2018). An example of a statistical test which is used to measure the dissimilarity between two dataset distributions is the Kolmogorov-Smirnov (KS). This statistical test will be used in this study in order to implement one of the concept drift approaches.

2.6 | Churn Prediction Models

The classification machine learning models that are used in this study are Extreme Gradient Boosting (XGBoost), Logistic Regression (LR), Random Forest (RF), Gradient Boosting Decision Tree (GB), Light Gradient Boosting Machine (LGBM). Furthermore, the Decision Tree (DT) model will be used in the data analytics aspect in order to identify any concepts occurring in the dataset. These models were chosen based on the evaluation paper and also based on the churn prediction model used by the industrial collaborator, in order to be able to compare the results.

2.6.1 | Logistic Regression

Logistic Regression has been quite actively used in churn prediction and especially in domains such as marketing, economics and telecom. LR is one of the more widely used models in the machine learning realm (Mahajan et al., 2015). This machine-learning algorithm has generally been used as a classifier by using the continuously bounded output and using a threshold to convert it into distinct classes. By continuously bounded, it is meant a varying value between 0 and 1 because the output from LR is a probability value.

The mechanics of LR are quite simple as this technique depends on the *logistic function* which is also referred to as *sigmoid function* and is defined as follows (Liu, 2017):

$$y(\mathbf{Z}) = \frac{1}{1 + e^{-\mathbf{Z}}} \tag{2.3}$$

where $\mathbf{Z}$ is the input, and y is the output. When given an input this function maps it to an output value between zero and one. Let us consider having a dataset input instance $\mathbf{X}$ containing n features $x_1, x_2, ..., x_n$. Assuming that each feature vector has a corresponding weight (coefficient) vector $\mathbf{W} = w_1, w_2, ..., w_n$ such that the input to the LR algorithm results to be (Liu, 2017):

$$\mathbf{Z} = w_0 + w_1 x_1 + w_2 x_2 + ... + w_n x_n = \mathbf{W}^T \mathbf{X} \tag{2.4}$$

where w_0 is termed as the bias. The output then becomes the probability of the label being the positive class given a certain input vector and defined as follows (Liu, 2017):

$$\hat{y} = P(y = 1|\mathbf{X}) = \frac{1}{1 + e^{-\mathbf{W}^T \mathbf{X}}} \tag{2.5}$$

which implies that LR is a probabilistic classifier. During training the vector of weights is adjusted having one goal in mind; to minimise loss and try to predict a positive sample as close to one and a negative sample as close to zero. The cost function that is generally using with LR is the log-loss function.

2.6.2 | Decision Tree

Decision trees have been quite extensively used in the prediction realm, especially in churn prediction (Bankar et al., 2016; Bin et al., 2007). These stem from graph theory, and as the name implies, their outcome resembles a tree-like structure. They are quite simple in nature and are effectively a sequential representation of all the possible decisions and the corresponding target. The entry point for such an algorithm is the root node, which leads to further branches forming, which also have an associated node. Each node will have a specific decision attached and depending on the choice made; different routes are traversed, which lead to other nodes. At the bottom end of the tree, there will be the leaf node termed as the terminating node, which gives the final predicted outcome (Liu, 2017).

Figure 2.7 illustrates a binary classification decision tree having two features as input being x_0 and x_1 respectively. Value L_0 is a decision boundary for x_0, and thus is the boundary segregating the input space. The decision in such case is to check whether the input x_0 is smaller or greater than L_0, whose value is obtained during the learning phase of the model. After the first split, the tree can be further subdivided into other regions because the new regions are independent. This is done by moving on and repeating this procedure to other input features, in this case, x_1. As an example, if one takes the upper boundary where $x_0 \geq L_0$ and subdivides according to other input feature, x_1 will end up with two regions having boundary L_1. This way, all the possible combinations are exhausted until no more partitioning can be performed. For instance, if $x_0 \geq L_0$ and $x_1 \geq L_1$ the prediction would be churn. This space partitions can be more easily visualised using Figure 2.8, which contains a series of nodes beginning from the root and ending at the leaves according to the decisions taking at each nodes using the boundaries learned during training.

There are several ways in which a decision tree can be created, such as by using the ID3 (Iterative Dichotomiser 3). ID3 adopts a greedy search for splitting the dataset using a top-down approach, C4.5 which is an upgrade of the ID3, Classification and Regression Tree (CART) and Chi-squared Automatic Interaction Detector (CHAID). All of the mentioned methods adopt a greedy search

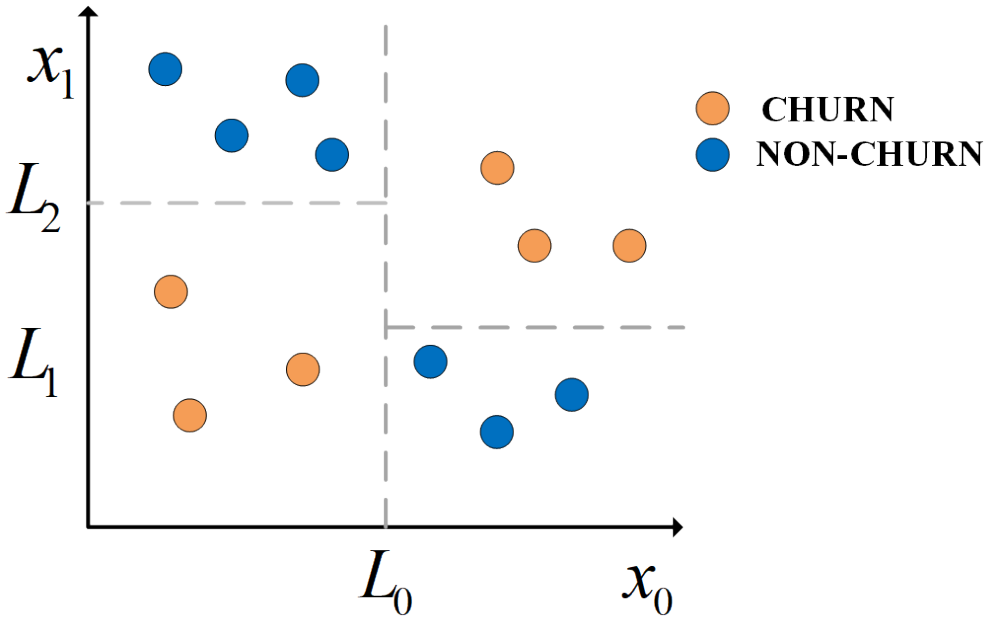


Figure 2.7: Decision Tree binary classifications regions.

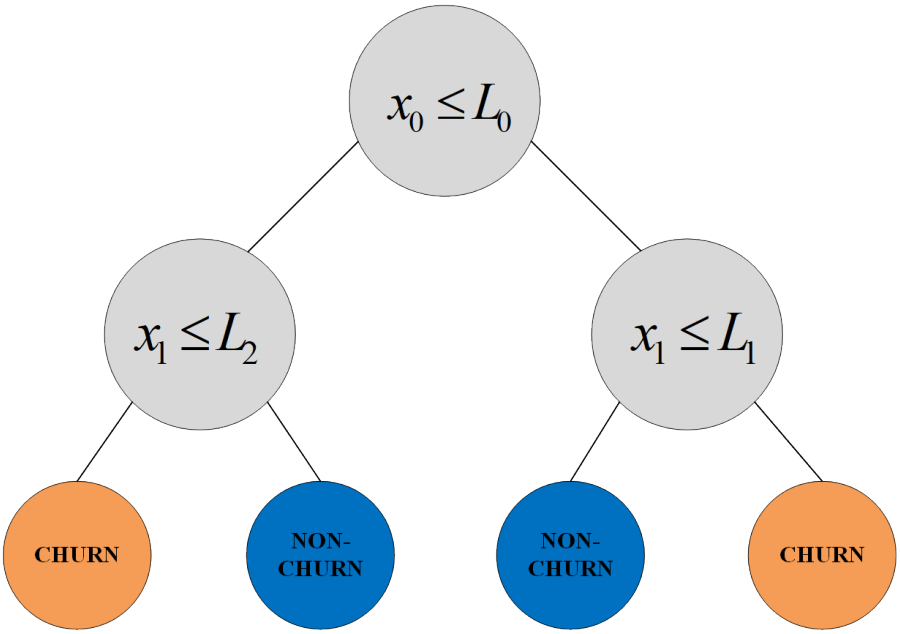


Figure 2.8: Decision Tree binary classifications paths.

based on local optimisation to grow and evolve the tree. In the CART implementation, the tree is grown using binary segregation by performing a greedy

search for the best combination from the input feature set with the aim of minimising loss. The process stops until either of two criteria are met being, the minimum sample count for the resulting node or the maximum depth of the tree is reached. These two criteria are present in order to prevent overfitting. Finally, the class of the leaf node is selected depending on the class with the most samples (Liu, 2017).

The splitting decision is based on a specific loss function and the value of information gain for each node. A loss function that is used is the Gini impurity (also known as index or coefficient) defined as follows (Liu, 2017):

$$Gini = 1 - \sum_{i=1}^{n} {p_i}^2 \tag{2.6}$$

where p_i is the probability for a specific class. It can be seen that Gini is a measure of disproportion in a sample and ranges from zero to one. A value of zero is an indication of similar elements, while a value of one indicates dissimilar elements.

DT has several advantages such as easy to understand and visualise as can be seen from the above illustrations. However, decision trees tend to tightly couple with the data causing overfitting, thus yielding highly dissimilar outcome on different test data. In order to tackle the variability issue inherent in DT ensemble methods were created, such as Random Forest (RF).

2.6.3 | Random Forest

Random Forest is part of the ensemble methods in machine learning and proved to be a beneficial model in churn prediction (Adiwijaya et al., 2014; Alamsyah and Salma, 2018). By ensemble method, it is meant that multiple models are used in order to achieve better predictive power. Thus, several classifiers are trained independently, each returning a specific class, which are then fed into a voting mechanism, and the final prediction is computed.

The main postulate behind RF is that when using several weak models, better performance is achieved when matched with one strong model. In RF, a weak model is assumed to be a DT which is trained on a subset of instances of

the whole dataset. The word ***Random*** in RF is there for two reasons which are crucial in this machine learning model:

- Randomly selecting samples of training data during DT construction
- Randomly selecting part of the whole feature vector during node splits

Once the DTs are trained, each will cast a vote for a class, and then majority voting is used to achieve the final class. Figure 2.9 explains the high level mechanics of RF. Effectively each tree uses a **random** sample of the training data; data sampling is carried out with replacement (also known as bootstrapping). This means that some data instances can be used more than once. The idea behind random sampling is that although the individual trees tend to have high variance as discussed in the previous subsection when combined into a group, the entire forest will have lower variance without affecting the bias. During node splits, a subset of the feature set is considered, and typically the number of features to be used equates to the square root of the total number of features available. Once each tree is trained, the response is gathered in a voting mechanism to produce the outcome. The concept of grouping the output of several trees are referred to as bagging, and thus the whole process is known as bootstrap-aggregating (Breiman, 2001; Koehrsen, 2018).

2.6.4 | Extremely Randomised Trees (Extra Trees)

The Extra Trees (ET) machine learning was proposed by Ernst et al. (2006) and in principle relates to RF. That is during selection at each node a random subset from the feature set is used to decide on the split. However, ET differs in the way each tree is built and does not make use of bootstrapping. This means that ET samples without replacement. Furthermore, the nodes are split randomly from the random subset and not on the best split as happens in RF hence the name randomised.

By adopting this technique, ET obtains accuracy that is on par with that of RF. However, ET is less complicated and less computationally intensive than RF because it does not involve optimisations used in searching for the best split.

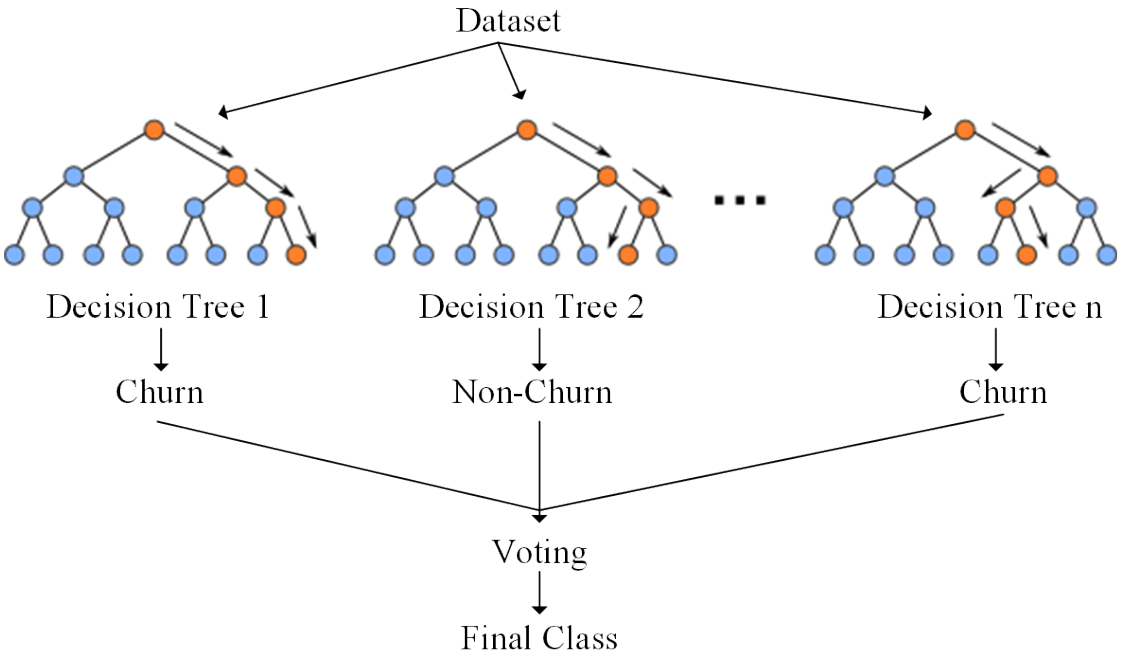


Figure 2.9: Random Forest mechanics.

Amongst several advantages, ET can learn a non-linear problem while using the least number of parameters possible and also this machine learning model tends to offer low variance (Geurts and Louppe, 2010).

2.6.5 | Gradient Boosting Decision Tree

Random Forest was an example of a bagging ensemble algorithm, but there exists another type of ensemble learning known as boosting. In boosting, sequential model training is performed rather than independent. A variant of boosting ensemble learner is Gradient Boosting Decision Tree (GB), which appeared in numerous churn prediction publications (Choi et al., 2017; Gregory, 2018). Effectively GB is an ensemble of sequentially trained decision trees. In such a boosting model, the mistakes in the classification of a certain model are learned in future models, and thus such models tend to converge faster (Grover, 2017). From a computational aspect, this model is intensive and is time consuming mainly due to the training of the individual decision trees in finding the best

splits. Figure 2.10 illustrates the differences between the various machine learning strategies discussed.

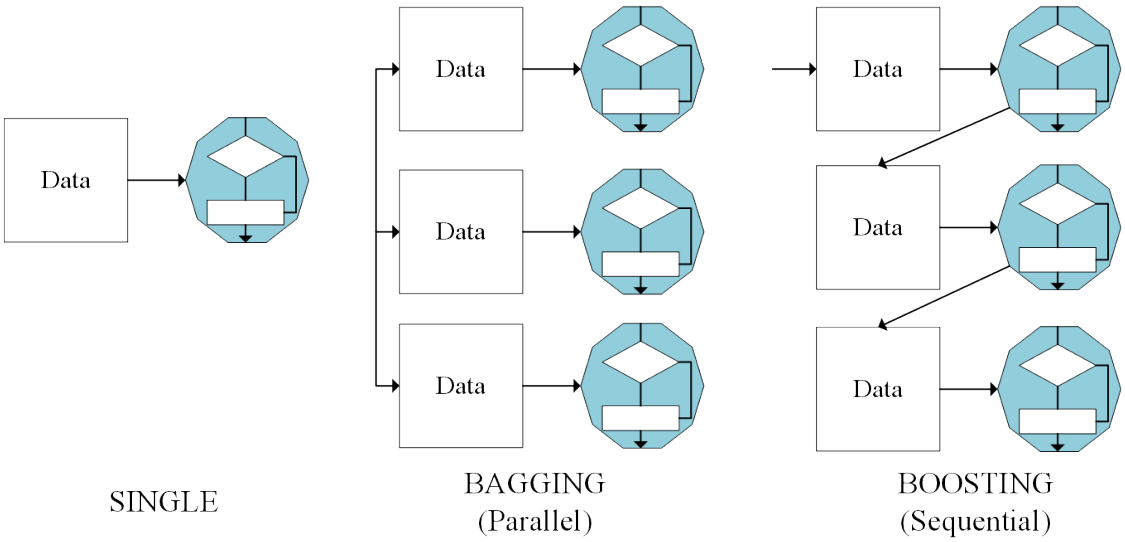


Figure 2.10: Different types of machine learning strategies.

2.6.6 | Light Gradient Boosting Machine

The GB model is not scaleable as it involves training many decision trees, which is time-consuming, especially when using large datasets. For this reason, a team from Microsoft proposed the Light Gradient Boosting Machine (LGBM) which makes use of two novel techniques being the Exclusive Feature Bundling (EFB) and Gradient-based One-Side Sampling (GOSS) (Chen et al., 2017). The latter is used to filter out part of the data having small gradients and use the other part to compute information gain. The reasoning behind this is that data instances having larger gradients are more important for information gain computation. Hence with the use of GOSS, accurate estimation is obtained for information gain with the benefit of shaving a lot of data volume. Furthermore, EFB, is used as a dimensionality reduction measure where mutually exclusive features are combined. By using this technique, the LGBM is more optimised when it comes to data fitting and thus is significantly faster than its predecessors. When

compared with GB, LGBM is up to 20 times faster while obtaining the same level or better in terms of predictive performance (Chen et al., 2017).

2.6.7 | Extreme Gradient Boosting

Another boosting type machine learning model is the Extreme Gradient Boosting (XGBoost). This model was developed before LGBM with the intention to optimise the length and complex training process inherent with boosting type models. As previously mentioned, finding the best split is a complex task, especially when using the greedy algorithm. Although using the greedy algorithm is the best option since all features are included, it is evident how one can start running into memory problems, especially on a large dataset. In order to counteract these drawbacks, XGBoost uses a sparsity-aware algorithm that is used for parallel tree learning. Dataset sparsity is common in machine learning, and in some cases, this sparsity is induced through the use of feature engineering techniques such as one-hot encoding. One-hot encoding increases the number of features, which means that in the case of the ordinary GB machine learning model, the training times will increase. In case of the XGBoost, adapting the sparsity-aware algorithm, the training times are kept to a minimum while performance does not suffer even on large dataset (Chen and Guestrin, 2016).

2.7 | Evaluation Framework

In order to evaluate predictive concept drift approaches, a different approach to that of a normal static dataset scenario needs to be used. This is so because models used in concept drift have the potential to evolve, and inherently the metrics will also change. A typical evaluation strategy in a static scenario is the cross-validation approach. However, such an approach is not appropriate for data that has varying trends or concepts because, through cross-validation, a portion of the dataset is used more than once, and chronological ordering is ignored. Several studies show that although concept drift machine learning approaches is picking up popularity, the evaluation aspect can still be considered as an open is-

sue due to several reasons. The first being data flow is continuous and not a fixed sample, the second being, that the machine learning model is constantly evolving rather than being static. Thirdly, data tends to be non-stationary instead of the typical stationary dataset. Albeit, there is no gold standard in assessing predictive performance in concept drift, especially in non-stationary environments as encountered in this study. Research shows that there are mainly two methods for evaluation being the holdout and the predictive sequential (Gama et al., 2009a,b; Shukla and Yadav, 2016).

2.7.1 | Holdout Evaluation

The holdout evaluation is also commonly known as the validation set approach or the classical evaluation approach, which was proposed to solve the overfitting problem (Shukla and Yadav, 2016). The dataset is split up into two non-overlapping portions, with a typical apportioning of 20% being used as the holdout test set. The model is trained on the remaining portion and never learns from the holdout set to gauge the model's ability to perform on unseen data. A downside of this approach is related to the fact that if the data distribution is skewed and there is a significant difference between the training set and the holdout test set, the resulting metrics will not be meaningful. Research shows that the majority of the customer churn prediction solutions employed this approach for evaluation (Shukla and Yadav, 2016).

2.7.2 | Predictive Sequential Evaluation

A better evaluation strategy for concept drift machine learning problems is that of Predictive Sequential, which is also known as Interleaved Test Then Train or simply Prequential. In this strategy, the model is continually testing future samples and if need be, depending on the training approach taken updated. Thus by using this method, the machine learning model is always testing on unseen data, and thus it is evident that no holdout set is required. Using this method, one can monitor the model's performance evolution across time, resulting in a smooth plot of the desired metric overtime where each result will become less

significant to the overall mean. Inherently, by using such method, one can evaluate the model's performance over time as the individual sample performance will be recorded (Gama et al., 2009a,b).

2.7.3 | Classification Evaluation Metrics

In order to evaluate and compare the performance of different machine learning models and training approaches, there are a plethora of metrics which one can use. When introduced to the data science domain, the most common metric that is mentioned is the accuracy metric (Hossin and Sulaiman, 2015). The main goal of a machine learning model is to achieve the best accuracy possible when the model is deployed on a production system. However, the accuracy metric should not be considered as a generic metric which applies in all machine learning scenarios. According to Burez and Van den Poel (2009), using such a metric on an imbalanced dataset leads to erroneous results and analysis, and emphasised that using accuracy in a churn prediction problem is not recommended. When using accuracy on an imbalanced dataset, a machine learning model can achieve an acceptable accuracy score but fail to predict real churners (Burez and Van den Poel, 2009). The work of Afshari et al. (2014); Khan and Rana (2019) shows that appropriate metrics for performance evaluation under class imbalance are Area Under the Receiver Operator Characteristic Curve (ROC-AUC) and Area Under the Precision Recall Curve (PR-AUC), with the latter being preferred, as the former tends to be optimistic depending on the ratio of the classes.

2.7.3.1 | Binary Classifier Evaluation Metrics

As mentioned earlier, in a binary classification task, two classes are present, the positive class (churner) and the negative class (non-churner). Hence when testing the model, the test set is comprised of positive (P) and negative (N) labelled instances. Using this labelling nomenclature, one can define several types of classification examples namely the True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN) (Burez and Van den Poel, 2009). TP denotes that the classifier correctly classified a positive instance, TN denotes that the

classifies correctly classified a negative instance, FP implies that the classifier incorrectly classified a negative instance as a positive one and FN means that the classifier incorrectly classified a positive instance as a negative one. Using these definitions, a Confusion Matrix (CM) can be created. A confusion matrix is one of the most straightforward and most intuitive ways to analyse the correctness of the model and is shown in Figure 2.11. A confusion matrix shows the values for the predicted and actual values, as shown in Figure 2.11.

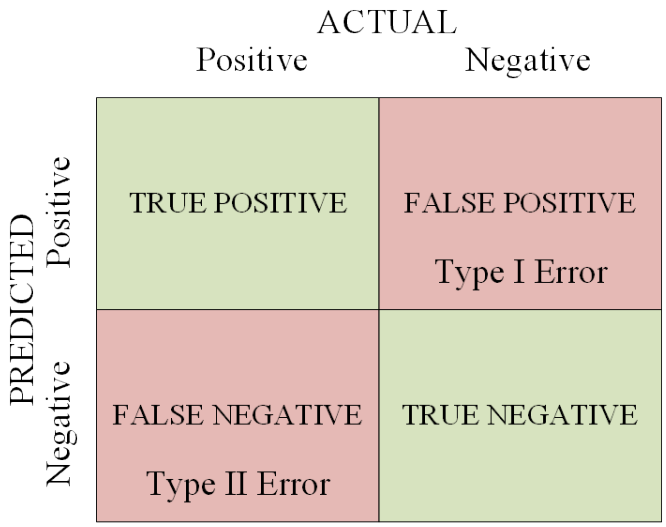


Figure 2.11: Overview of the Confusion Matrix.

Using these definitions, one can note that TP + FN = P and similarly TN + FP = N which leads to several definitions commonly used in a classification task and shall be used in this study such as Accuracy, Specificity, Precision, Recall, F-measure (FM), False Positive Rate (FPR), Area Under the Receiver Operator Characteristic Curve (ROC-AUC) and Area Under the Precision Recall Curve (PR-AUC).

The accuracy metric measures the overall correct predictions and is defined as follows (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.7}$$

The specificity metric measures the ratio of the correctly predicted negative examples from all the actual negative examples and is defined as follows (Flach,

2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$Specificity = \frac{TN}{TN + FP} \tag{2.8}$$

The precision metric measures the correctly predicted positive examples from all the positive predictions available and is defined as follows (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$Precision = \frac{TP}{TP + FP} \tag{2.9}$$

From the equation (2.9), it can be noted that a model that has zero FP has unity precision. Precision tries to answer how many of the positively predicted entries were predicted correctly

The recall metric measures the fraction of positive instances that are correctly classified and tries to answer how many of the actual positives are predicted correctly and is defined as follows (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$Recall = \frac{TP}{TP + FN} \tag{2.10}$$

From the equation (2.10) it can be noted that if a model does not have any FN leads to unity recall. This metric is also known as Sensitivity, Hit Rate or True Positive Rate (TPR) (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015).

The F-measure also termed as F1 score, or F Score is the harmonic mean using precision and recall metrics, or in other terms a balance between recall and precision, which is defined as follows (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$F - measure = \frac{2.Precision.Recall}{Precision + Recall} \tag{2.11}$$

The False Positive Rate is the complement of True Positive Rate (TPR) or Recall. It aims to discern the incorrectly predicted positives with respect to the actual negatives as is defined as follows (Flach, 2019; Hossin and Sulaiman, 2015; Zheng et al., 2015):

$$False\ Positive\ Rate = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{2.12}$$

To explain the ROC-AUC metric consider a probabilistic classifier that outputs a probability for each instance. Using a certain threshold t this classifier can be converted into a binary classifier as the probability output by the classifier is categorised into either one of the two classes according to the threshold. If for each threshold the TPR and FPR are plotted, the resulting plot is known as the Receiver Operator Characteristic (ROC) Curve. The area under this plot gives an indication of the model's performance across various classification thresholds, and thus it is invariant to the classification threshold. The ROC-AUC is worked out by calculating the area under the graph, and the result is bounded from zero to one. It can be deduced that a random classifier will have a ROC-AUC value of 0.5 while a perfect classifier attains a value of one (Burez and Van den Poel, 2009). The ROC curve of three different classifiers is illustrated in Figure 2.12. The classifier shown in red is labelled as the random classifier since it has no predictive power as it is not able to discern the different classes. The perfect classifier is shown in green and is represented as a step function. This classifier correctly predicts all instances. Finally, there is the typical classifier shown in the blue curve and is placed in between the perfect and the random classifier.

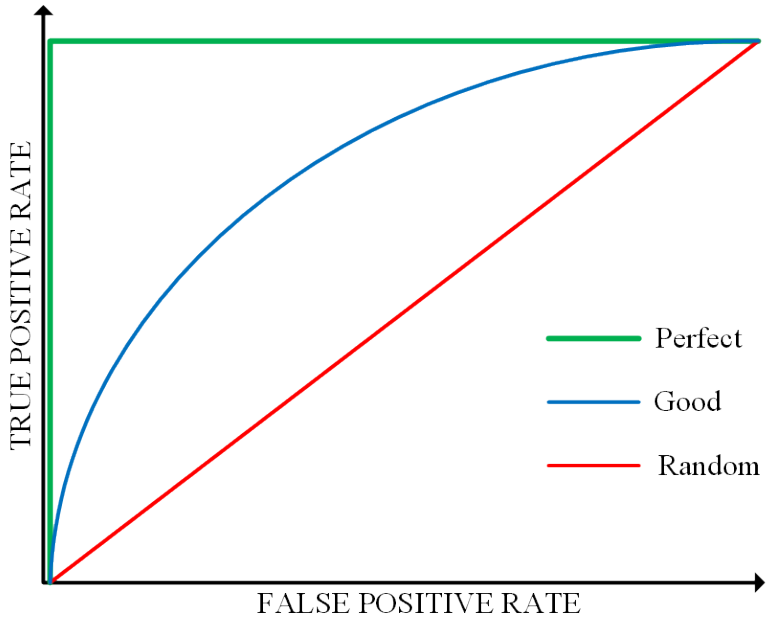


Figure 2.12: ROC plot for three classifiers.

The Precision Recall (PR) curve is a plot of the Precision versus Recall. Similar to the ROC plot, different thresholds are used. However, it is focused on predicting the positive class, which in our case is the churn class. The area under the PR curve results to be the PR-AUC which is also referred to as Average Precision (Zheng et al., 2015).

2.8 | Summary

This chapter served as a deep dive into the theoretical background of this work, and it will serve as a reference to the various areas and methods that will be used. First, an overview of CRM was given in order to accustom the reader with the problem to be solved. Concept drift was explained thoroughly, and several drift mechanisms were discussed. Then the metrics that shall be used during the evaluation were presented. The chapter closed by explaining the mechanics of different machine learning models that shall be used in this study.

3

Literature Review

Without doubt churn prediction has been extensively and still is actively researched due to being a key issue in various domains such as telecommunications, automotive industry, banking services, energy sector and internet service providers amongst others (Mandák, 2018). This chapter discusses how past literature tackled the problem of churn prediction, in particular, what machine learning models were used and the results obtained. Furthermore, this chapter presents the work of Kim *et* al., which will be used to compare the performance of concept drift mitigation techniques.

3.1 | The Evolution of Churn Management

The notion of churn management forms part of Customer Relationship Management (CRM) and is tightly coupled with customer retention. CRM is not a novel concept, as customer centred systems started appearing in the late 20^{th} century with the advent of database marketing (Davies, 2013; Writer, 2013). Back then, customer entries in the database were used to tailor the customer's marketing communications and this formed part of the direct marketing strategies. Database marketing allowed companies to follow the customer's transaction logs and direct their marketing strategy accordingly (Dawson and Minami, 2008).

Following the internet growth, new companies began to emerge, which con-

sequently led to market saturation for several domains, especially telecommunications. More companies led to more competition because the customer can choose between the competing companies (Gholamian et al., 2010). At a point in time, companies assimilated this increase in competition and started to rethink their marketing strategies. These marketing strategies started to focus more on customers with a high probability of churning to minimise churn (Chu et al., 2007). This incentivised several researchers to conduct studies on the negative impact of customer churn in different domains. One industry in which companies are more prone to customer churn is the telecommunication industry because the end-user can easily switch companies. This leads to a typical mobile phone service provider incurring a loss of more than four percent of their monthly customer base resulting in the reduction of the total yearly revenue by millions of dollars (Chu et al., 2007). Other researchers point out that the issue of customer churn is an ever-increasing one and that in order for a company to be profitable, it needs to establish long term customer relationship (Swait and Sweeney, 2008). This reasoning is backed by other publications which support the idea that customer retention is better than trying to lure new customers (Eriksson and Vaghult, 2000). Hence the claim by Farías et al. (2008), Bendapudi and Berry (1997) and Andreassen et al. (2001) that a one percent improvement in customer retention rate will boost the firm's value by five percent. Similarly, Reichheld and Sasser (1990) mentioned that if customer retention increased by five percent, the company's profits would increase in the range of 25% to 85%.

It is evident that the company's main target is to increase customer retention. Several studies support this and explain that in order to thrive in the industry, customer retention should be the number one priority (Larivière and Van den Poel, 2004; Lin and Xu, 2008). Researchers Buckinx and Van den Poel (2005) explain that it is useless for a company to invest exorbitant marketing budget in attracting new customers and that the company should put more effort in customer retention because in the long term loyal customers will generate more revenue. This is particularly visible in the telecommunication domain because typically such companies offer a multitude of services. If the customer is retained for a more extended period, the company will have a larger time frame in which it can offer more of its products, which typically lead to more revenue (Babad

et al., 2008). This is supported by the claim made by (Eriksson and Vaghult, 2000) that "customer retention leads to increased sales and reduced marketing costs compared to selling to new customers".

If the marketing strategy of a company does not prioritise customer retention over customer acquisition, it is difficult to survive in a saturated market. This can be achieved by employing a churn management system. Furthermore, although long term customers contribute to increasing profit, customers that churn impose irrecoverable damage to the company's track record (Gans, 2000). The choice of giving more importance to customer retention rather than to customer acquisition is crucial when having both a large customer base and harsh competitors because at this point it is challenging to secure new customers and easy to lose customers. When dealing with churn management and customer retention, the partitioning of customers needs to be evaluated carefully because there are several types of churn.

According to Capota and Lazarov (2007), Naz et al. (2018) and Shourabizadeh and Yigit (2017) the types of churn can be split into two main groups being voluntary and involuntary. The voluntary group is further split into two other groups being deliberate churn and incidental churn. A deliberate churner, also known as active churn is when a customer chooses to make a switch to another company and quit the current company. This may happen due to lack of customer satisfaction. An incidental churner, also known as rotational, occurs when the customer quits the contract without the intention of moving to another company. This may occur when there are reasons that inhibit the customer from renewing the contract such as due to demographic reasons (customer moved away) and financial problems amongst others. This type of churn accounts for only a small percentage of voluntary churn, and since it is mainly linked with the customer having financial problems, it is also termed financial churn (Burez and Van den Poel, 2008).

Finally, the last type, involuntary or passive churner which occurs when the customer misbehaves and the company decides to disassociate itself from the customer. Some of the reasons that could cause a company to do this are when the customer abuses from the service offered. Another example would be in the case of iGaming. Consider that a customer shows signs of gambling addiction,

bonus abuse or portrays fraudulent behaviour. In this case, the iGaming company has a mechanism which stops the customer from using the online services, and thus, such customer will be considered as involuntary churn.

3.1.1 | Churn Drivers

Churn management is concerned with voluntary deliberate churn, which is highlighted in the previous section. The driving force behind a customer's move to another company is generally due to unsatisfactory experience, and the reasons vary from one domain to another. A study carried out by Adebiyi et al. (2016) elucidates several important factors and reasons why churn happens, in particular, in the telecommunication industry.

- **Cost (Price)**: This refers to the amount of money a customer has to pay in order to acquire the company's service. Customers are always seeking for low service price and thus churn depending on the price. If the price of the service is higher than that of competitors, customers are ready to switch. Adding to this, if customers witness constantly changing fees and lack of transparency, increases the probability to churn. Hence there is a tendency for companies to lower their service cost in order to attract new customers and reduce churners (Anuwichanont, 2011). This is supported by several researchers, who claim that customers typically set a price reference and if they perceive a company's service price to be comparable to this reference, do not churn. This explains the importance of fair service valuation and that higher service price affects customer purchases (Jiang and Rosenbloom, 2005). In the case of telecommunication services when a company does not adopt switching costs, customers tend to churn because they would not incur any cost when joining new competitors. A churn phenomenon occurred back in 2003 when a new law was introduced in the US. The law was about number porting, which means that when switching mobile phone providers, the customer could carry the existing number; before this law, when switching providers entailed registering a new mobile number. According to Eshghi et al. (2007), when this law was

put into effect, 12 million customers churned to a different mobile phone service provider. This phenomenon effectively spurred further the ever-present struggle of customer retention. According to Chu et al. (2007), the price accounts to 47% of churners in the telecommunications domain, and points out that this is the main reason why churn happens.

- **Satisfaction**: Customer satisfaction is another critical factor and is explained as the difference between the customer's expectations and what was actually received. It deals with the customer's reaction after using the service. Thus clearly, higher satisfaction enables the customer to stay and thus do not churn (Hejazinia and Kazemi, 2014). From a high-level viewpoint, satisfaction groups several other churn drivers such as service quality. Furthermore, according to Chu et al. (2007) satisfaction accounts for 35% of the churners and hence is the second-largest churn contributor. Since it encloses several other churn drivers, customer satisfaction is prone to suffer from the quality of the service being offered, such as billings issue and lousy customer service. Literature reveals that customer satisfaction is a severe problem to solve and iterates that customer satisfaction needs to be at the top of the marketing agenda in order to succeed (Jeong et al., 2004). The notion of customer satisfaction, although different, links with customer loyalty, in fact, a publication by Bruhn and Grund (2000) showed that in the telecommunication industry customer satisfaction construes almost 100% of loyalty. Loyalty is attributed to those customers that exhibit optimistic perspective towards a service, product or brand and this can be manifested in several ways such as recurring purchases, indifference to competitors' offers and trying different products a company offers (Rundle-Thiele, 2005).

- **Quality**: The quality of the service being offered is also a significant factor. Research shows that service quality is the distinction between the actual service offered and the advertised service. In a way, service quality collects the customer's feeling after using the company's service (Parasuraman, 1998). Service unreliability and an inappropriate way in which the service was delivered would push customers away. When a customer files

a complaint, an adequate and quick response is awaited for. Thus lack of responsiveness and poor service quality would lead to churn. A company shall ascertain that service billing is carried out in a correct way without any errors as this would lead to churn. As stated in literature, the reasons aforementioned can be grouped into two subtypes of quality being technical quality and functional quality. The former reflects the service's performance while the latter describes the customer's interaction with the service (Qin, 2012).

- **Privacy and Security**: Customers may churn for privacy concerns because customers do not expect their personal information to be mishandled and it is the company's responsibility to keep customer data safe. Lack of privacy and security instils anxiety and fear in customers due to several concerns such as the idea of being monitored during a call in case of the telecommunication industry (Hejazinia and Kazemi, 2014).

- **Innovation and Technological Advantage**: Customers are always searching for new features, and they are ready to churn if the product lacks the features that they need. If competitors offer new enticing products which include new technology at the same price point, customers will make the switch. Additionally, the introduction of new competitors would tempt customers to make the switch because typically new market entrants offer better package deals which include the latest technology. Studies show that companies offering technologically superior products at a lower price point magnetise customers (Hejazinia and Kazemi, 2014).

From the aforementioned churn drivers, it is evident that a churn management system should not consider the whole customer base for customer retention because if a customer misbehaves and does not pay, that customer is not worth retaining, implying that some of the customers are not worth holding on to. Secondly, customer retention involves a considerable amount of money. According to Qin (2012) in a study on airline business pointed out, that it is useless to include loyal (long term) customers in the retention strategy, because they have no intention of churning. The element of understanding customer needs is

essential, and according to Liu and Shih (2005), in order to increase customer retention, companies were forced to develop marketing strategies according to the customers' needs. Thus moving away from the traditional approach of trying to increase sales by targeting customers blindly.

3.2 | Churn Management Systems

According to Burez and Van den Poel (2007), there exist two types of approaches for churn management systems being reactive and proactive. As the name implies when a company adopts a reactive approach, the customer is awaited to terminate the relationship with a company and therefore churn; this is also known as a passive churn system. On the other hand, in a proactive approach, the company is on alert and tries to actively determine customers who are likely to churn, hence adopting an active approach. The company takes the initiative by incentivising these customers to reduce the probability of churn. However, such approaches employ churn prediction systems, and thus the proactive approach can waste resources if these churn prediction systems are inaccurate. Therefore it is imperative to build a churn prediction system as accurate as possible in order not to waste money on loyal and long term customers (Adebiyi et al., 2016; Burez and Van den Poel, 2007). A study carried out by Mandák (2018), described the three main dimensions that a churn management system should be able to answer. These are the following:

- **WHO**: The system should be able to pinpoint the customers that are on the verge of churning.
- **WHEN**: It should indicate when the churn will happen.
- **WHY**: The system shall identify the reasons of why the customer churned.

During the design stage of a churn management system, the researcher has to keep in mind the pertinent factors that affect churn in the domain the study is carried. For instance, in a study carried out by Ardabili and Keramati (2011)

when considering customer dissatisfaction for an Iranian mobile network company used features such as service failure rate, complaints and customer lifetime. When dealing with the service usage factors, features such as transaction count, transaction value, contract information and payment history were considered. Finally, there are various customer features to include in the customer-related variable factor. Typically the customer's demographic variables are included such as age, nationality, address, level of education, income, gender and locality (Clemes et al., 2010). A salient point highlighted by Dahiya and Talwar (2015) emphasises that for a proactive churn strategy to work well, ample time needs to be spent in the creation of pertinent features. Typically the above-mentioned customer factors can be grouped in three generic factors as follows (Mandák, 2018):

- **Customer Demographics**: This factor contains the demographic features of the customer such as name and age amongst others as mentioned in customer-related variable
- **Customer Behaviour**: Containing features about the customer's service usage patterns.
- **Customer Perceptions**: Involving features that are extracted from, for example, customer surveys and thus monitor the customer's satisfaction concerning the quality of the service offered.

In this discussion, it is worth mentioning the Recency, Frequency and Monetary (RFM) feature model, which includes the customer features previously discussed amongst others. Recency is one of the three quantitative feature category offered by this framework and typically refers to the interactivity latency of the customer with the company. For example, it refers to the last time a customer made a purchase from a company. The Frequency part of RFM refers to the customer's frequency of activity; that is looking back at the customer's activity log how often did a purchase happen. The last part of RFM is Monetary, which relates to the amount spent by the customer. In relation to this study, the RFM feature model is used to obtain features related to the number of bets players (frequency), the last activity of the player (recency) and the amount of

deposit put forward (monetary) amongst others. Figure 3.1 illustrates the discussed generic feature model, which shows the main feature groups which affect customer churn. Hence such feature groups should be prioritised during feature engineering to develop the most appropriate feature set to create the best performing machine learning churn prediction system.

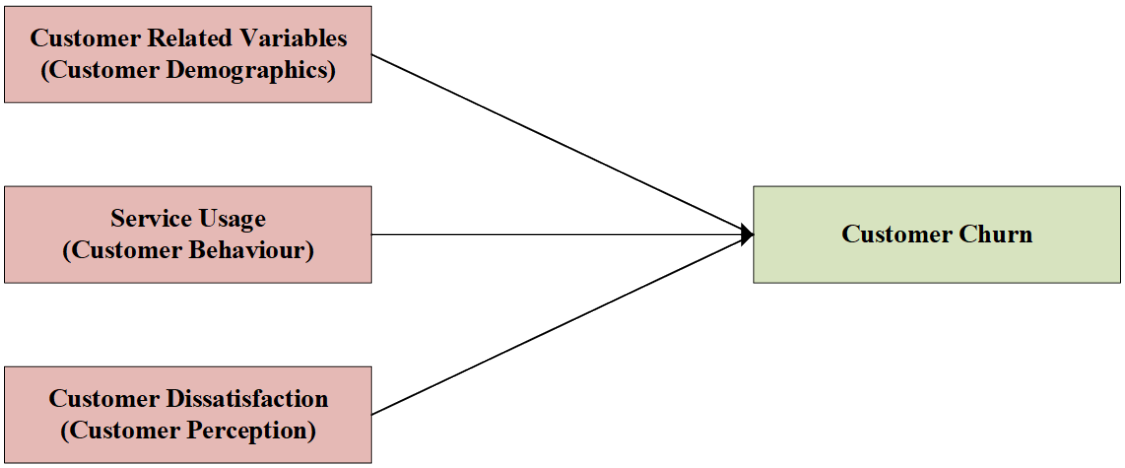


Figure 3.1: Churn model concepts. Adapted from Ghaneei et al. (2016)

3.3 | Feature Engineering

Different data and different feature combinations hold different predictive power, and it is crucial to discern the appropriate features for churn prediction. Typically the standard features that are domain agnostic are those pertaining to service usage as studies have unanimously agreed on this. For example, in the case of churn in the Internet Service Provider (ISP) domain usage data was used to derive a number of features (Liu and Ng, 2000). Similarly, usage data proved its importance when investigating customers' behaviour during website navigation (Ghose et al., 2003). Other literature suggests that the customer's past purchasing data is adequate in predicting what the next product the customer will buy or if there will be any repeated purchases for a particular product (Donkers and Verhoef, 2001) is. When it comes to predicting customer behaviour the preferred variables are recency, frequency and monetary which form the RFM fea-

ture model as they hold a significant amount of predictive power and have been used in many domains such as insurance and gambling industry amongst others (Alhaery and Suh, 2016; Donkers and Verhoef, 2001). From this discussion, it is clear that when choosing and creating features, one has to keep in mind the domain knowledge as much as possible.

3.4 | Feature Selection

Feature selection occurs during the data analysis and modelling phase. The outcome after performing feature selection is a set of features that are the most appropriate for describing the prediction problem and that best matches with the model mechanics. According to literature, feature selection involves two stages, the searching phase and the evaluation phase (Baesens et al., 2018; Bebis et al., 2004; Mwadulo, 2016).

The search phase can be broken down into three types being optimal, heuristic and randomised. Optimal search is an exhaustive and greedy search method and thus is not recommended when having a large number of features. However, other options avoid the exhaustive route such as the branch and bound algorithm as described by Bebis et al. (2004). Some of the most used heuristic selection techniques are the Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS). In the former, the search phase begins with an empty feature set, and on each iteration, the feature with the highest score is added. In the latter the opposite happens, where the iteration process begins with the full feature set and after each iteration, the feature that had minimal impact on the predictive power is dropped (Mwadulo, 2016). Finally, the randomised feature selection type adopts sampling techniques as well as probabilistic one to select features. A type of randomised feature selection approach is the Genetic Algorithm (GA). GA adopts the principle of survival of the fittest. On each iteration, the GA evaluates what is called the fitness function, and the features with the highest probability are selected (Bebis et al., 2004; Mwadulo, 2016). According to Pendharkar (2009) when using GA to explore the feature space customer churn prediction accuracy was improved.

The other stage in feature selection is the evaluation phase which can be split into two groups being the filter and wrapper methods. The distinctive trait between these two types is whether the feature subset evaluation is carried out using a learning algorithm incorporated in the classifier or not. In filter type, the feature set evaluation is external to the classifier design, while the wrapper type using a learning algorithm that is consolidated with the classifier. According to Bebis et al. (2004), filter methods are more computationally efficient since the best features can be quickly evaluated externally to the classifier, however, may lead to erroneous feature selection particularly in the case where the classifier is feature sensitive.

3.5 | Churn Prediction Approaches

The majority of research on churn prediction is based on supervised machine learning algorithms. However, there are unsupervised machine learning options and also other research which involved heuristics and advanced data processing techniques (Ahmed and Linen, 2017).

3.5.1 | Supervised Approaches

According to Bankar et al. (2016), the most commonly applied supervised machine learning models in churn prediction are logistic regression and decision trees. In one of their publications, Bankar et al. (2016) applied logistic regression and decision trees to predict customer churn in the telecommunications industry. They proposed a system which can be freely used by telecommunications companies and that allows several options to the primary users. These options were termed views, and in one of the views, the resulting output of logistic regression and decision trees is given. The end-user will then be able to give the churn prediction system a dataset in order to perform the learning process on both models, all of which is done through a web interface. The authors point out that the best model is not the one that gives the best precision score. However,

it is the one that furnishes insights that help in preventing churn with respect to algorithm efficiency on the specific dataset Bankar et al. (2016).

Hwang et al. (2004) produced work on customer segmentation and customer lifetime value in the wireless telecommunications industry. Customer segmentation and lifetime value, are closely linked with customer churn because the lifetime value depends on customer loyalty which is defined as (1 - churn rate). The study shows the importance of the churn rate value when considering customer cultivation and points out that past work did not consider churn rate when studying customer value. As for the machine learning techniques used logistic regression, decision trees and neural networks were included. Although the decision tree model achieved slightly better accuracy results than logistic regression and neural network, the study reports that decision trees cannot be deemed more superior in this scenario.

Motivated to solve the real-world recurring problem of customer retention, researchers Liu and Ng (2000), embarked on a research journey to study such an issue. The study involved a collaboration with a large service provider company whose details were not disclosed due to confidentiality and sensitivity reasons. Since customer retention involves the requirement of a highly accurate feature set, the researchers chose to use the C4.5 decision tree. Much emphasis was made on the accuracy of the feature selection process and using a C4.5 decision tree, the attributes that give a high accuracy are the features that are most relevant and significant.

Even though logistic regression and decision trees are amongst the most popular supervised machine learning techniques in customer churn literature, does not imply that they are perfect. In fact, Bock et al. (2018) point out some drawbacks of these models and highlight that decision trees tend to have difficulty in handling linear relations between input features. In contrast, logistic regression tends to be sensitive to input features interaction. For this reason, the authors proposed a new hybrid classifier named the Logit Leaf Model (LLM), aimed for better classification. The reasoning behind the proposed model was that using several models for different data segments rather than on the whole dataset will lead to better predictive power, while also keeping model comprehensibility present in the leaves. The developed model consists of two phases being

segmentation and prediction. In the former, a decision tree is used to identify the customer segments, while in the latter phase, a logistic regression model is applied for every leaf in the tree from the first phase. Results showed that the LLM outperformed the basic logistic regression and decision trees and performed equally well to well-known ensemble methods such as random forests and logistic model trees. The performance improvement when using LLM was that of five percent when compared to logistic regression.

Agarwal et al. (2016), also contributed to the area of customer churn specifically in the telecommunications industry domain. For analysing churn, service usage and customer-related information were used. With regards to supervised machine learning techniques, decision trees and their ensembles were used. The ensembles were limited to random forest and gradient boosted trees. Due to having large amounts of data, big data tools such as Apache Spark were used to train and test the models. Having a distributed architecture allowed the authors to perform grid-based hyper-parameter optimisation in a relatively short amount of time, which resulted in achieving better results. Results favour the models employing the grid search technique with some models having a five percent increase in predictive performance when compared to the result without the hyper-parameter optimisation. Also, the ensemble methods outperformed the base models with gradient boosting and random forest achieving an accuracy of 96.78% and 95.31% respectively, while the decision tree managed an accuracy of 89.99%.

Apart from the above mentioned supervised machine learning techniques, the area of soft computing has also been applied to churn prediction. By soft computing, it is generally referred to as a machine learning model related to neural networks. Although not widely adopted in the area of churn prediction researchers Bharadwaj et al. (2018) managed to use a Multilayer Perceptron (MLP) neural network to predict churn in the telecommunication industry. The MLP model had three hidden layers and achieved an accuracy of 94.19%. The researchers also used a logistic regression model for evaluation purpose, which managed to achieve an accuracy of 87.52% after regularisation. The evaluation was done using a train-test split and did not use cross-validation techniques which might prove otherwise if tested. In another publication, Dong et al. (2018)

introduced a novel approach to churn prediction termed product-based Recurrent Neural Network (pRNN). The pRRN is an RNN model which includes long short term memory units and thus able to ingest and learn sequential customer data. Results show that using such method significantly outperforms a regular long short term network.

3.5.2 | Unsupervised Approaches

Unsupervised approaches include models that do not require a labelled dataset and is mainly comprised of cluster analysis techniques. Clustering techniques have been used in conjunction with supervised machine learning techniques as a data pre-processing mechanism. Liu and Zhuang (2015), proposed a pipeline that uses a decision tree as the prediction technique and also uses a customer segmentation technique before the decision tree. In customer segmentation, as the name implies the customer is binned in a specific group depending on behaviour, demographics and preferences. In the study, K-means was used as the clustering technique from segmentation since K-means can cluster a large dataset rapidly and effectively. Customers were segmented into three levels, high, medium and low; then each cluster was forwarded to a decision tree for prediction. For evaluation purposes, the study compared the decision tree with logistic regression and artificial neural network, with results favouring the decision tree. The authors emphasise that if customer segmentation (clustering) is performed before using a supervised learning approach is beneficial because the upstream algorithm is trained on a particular type of data and hence is able to discern churn more efficiently and accurately (Liu and Zhuang, 2015).

A study carried out by Franciska and Swaminathan (2017) compared various clustering techniques in the churn prediction domain using KNIME software. One of the clustering techniques involved was hierarchical clustering. It was found that an advantage hierarchical clustering has over, for example, k-means is that at the beginning of the process there is no need to specify how many clusters are needed; it starts with the whole dataset in a single cluster, and the dataset is clustered on each iteration effectively ending up with N clusters. Despite having this advantage, hierarchical clustering does not scale very well when in-

troduced to large datasets. Density-based spatial clustering of applications with noise (DBSCAN) clustering technique was also used, and it was pointed out that it does not require predefining the number of clusters required and hence tend to suffer from the same issue of hierarchical cluster. That learning tends to be lengthy when subjected to large datasets having high dimensionality (Franciska and Swaminathan, 2017).

Dannoun et al. (2015) contributed to the area of customer churn by using hybrid modelling techniques. The hybrid nomenclature refers to a two-stage pipeline, being the clustering phase and the prediction phase. For the first stage, the authors tried three different clustering techniques being k-means, hierarchical and self-organising maps (HSOM). HSOM is an unsupervised machine learning technique, that is comprised of one to three-dimensional lattice of units where weighted links interconnect the units. In HSOM vector quantisation is used for training where the entire space is divided into boundaries. Results showed that the combination of k-means together with a multi-layer perceptron outperformed the other clustering techniques achieving an accuracy of 97.2% while 94.8% and 95.9% for hierarchical and HSOM respectively (Dannoun et al., 2015).

3.5.3 | Other Approaches

From the literature, several other churn prediction approaches were conveyed. In particular, the use of a Hidden Markov Model (HMM). In a study carried out by Faltings et al. (2015), the problem of predicting customers on the verge of churning considering the last 24 hours was tackled by using an HMM. HMM works on the principle of state independence where the activity at time $t + 1$ depends only on the state present at time t. The result showed that HMM performed equally well as the traditional supervised learning techniques mainly logistic regression, neural network and support vector machine however the authors emphasise that HMM offer several advantages when it comes to real-world implementation. This is so because the storage and computational requirements for HMM are low (Faltings et al., 2015).

3.6 | Class Imbalance Approaches

Depending on the churn prediction case, it could be the case that churn is frequently occurring while it may be a case of a rare event. This leads to having an imbalance between the churners and non-churners, which leads to misleading results. According to Burez and Van den Poel (2009), there are several ways to tackle class imbalance being by choosing appropriate evaluation metrics, by adopting cost-sensitive learning, by using sampling techniques and by using boosting methods. When dealing with evaluation metrics, the authors suggest using ROC analysis in order to assess accuracy independent of the classifier and or thresholds that may be present. Cost-sensitive approaches exploit the fact of class rarity, and when encountering false negatives, a higher cost is given. There are many sampling options such as under and oversampling; undersampling refers to data set reduction as the majority class is reduced to the same amount of the minority class while in oversampling the minority class is reproduced to equal the majority class. Finally boosting techniques tend to improve prediction accuracy since during each stage, weight is added in order to reduce the error. The study showed that undersampling could lead to improved accuracy when used with ROC-AUC (Burez and Van den Poel, 2009).

In a publication by Adnan et al. (2016), several imbalance algorithms were also compared. These consisted of Mega-trend Diffusion Function (MTDF), Synthetic Minority Oversampling Technique (SMOTE), Adaptive Synthetic Sampling (ADASYN), Majority Weighted Minority Oversampling Technique (MWMOTE), Immune Centroids Oversampling technique (ICOTE) and Couples Top-N Reverse k-Nearest Neighbor (TRkNN). From the listed techniques the MTDF option performed best. Other research combined sampling techniques with cost-sensitive learning models such as Weighted Random Forest (WRF), which was proposed by Adiwijaya et al. (2014). Although using a cost-sensitive model, it was noticed that with the introduction of combined sampling, the F-measure was improved (Adiwijaya et al., 2014). Researchers Azhar and Hanif (2017), also contributed to the area of class imbalance problems where they tried to use three different sampling techniques (oversampling, undersampling and SMOTE) to solve the imbalance in customer churn prediction. Results showed that over-

sampling achieved better performance contrasting the results obtained by some researchers that have been previously discussed (Azhar and Hanif, 2017).

3.7 | Concept Drift Approaches

The area of concept drift detection and mitigation is mostly dominated by literature in the financial domain, specifically for fraud detection. When it comes to customer churn prediction concept drift approaches are somewhat novel and rare to find in literature.

According to Alippi et al. (2015), concept drift is tackled by using a classifier which is trained using a sliding window on the whole time-ordered dataset and also by using an ensemble approach. The study by Alippi et al. (2015) shows how the sliding window approach trains a model every day while the ensemble approach apart from training the model on every day the final prediction is supported by previously trained model hence forming an ensemble to come up with the final prediction of classifying the transaction as fraud or not. Such an approach is also known as a passive concept drift approach as the model is updated at regular intervals without evaluating if concept drift was really present. There exists another approach which is known as the active approach where the model is updated only if a data shift occurs; the active approach requires a detection mechanism through the use of a statistical test to gauge the dissimilarity (Bai et al., 2018).

Machado and Ruiz (2017) propose a concept drift aware churn prediction system based on mobile application usage. The proposed solution consists of two stages the data stream clustering stage and churn prediction stage. The first stage by collecting all the data and performing the necessary preprocessing. This is carried out using specific window length where the data in each window is separately processed. Furthermore, clustering is used when the customers in each window are mapped to a specific cluster group; hence customers exhibiting same behaviour are grouped. The second stage is used to detect concepts and also predict incoming customers. Thus this stage aims to classify if a customer is likely to abandon their mobile or not, which is carried out by com-

paring the customer's behaviour with the clustering analysis resulting from the first stage. This approach is somewhat different from the traditional fraud concept drift detection approach; however, results obtained by Machado and Ruiz (2017) showed an accuracy of 87%.

A recent study of Fong et al. (2020) which regards customer churn prediction in the telecommunication industry proposed a churn prediction model that can handle data drifts. In the study, the customer data stream has a class imbalance and thus SMOTE is used to balance the dataset. The next step in the process is the concept drift detection. This was employed through the use of the Page Hinkley (PH) test. Such test is used to test abrupt signal detection and hence concept drift. If the PH reports the change as significant, the classifier is rebuilt to cater for the new concepts. Using this proposed framework resulted in a three percent reduction in error rate.

3.8 | Reviewing the Work of Kim *et* al.

This section gives a thorough analysis on the review and investigation that is carried out on the research paper by Choi et al. (2017). The paper "Churn prediction of mobile and online casual games using play log data" by Choi et al. (2017) is the benchmark paper chosen in this dissertation on which the concept drift analysis and approaches are applied. This benchmark paper by Choi et al. (2017) is selected based on several reasons being firstly dataset availability, as the dataset used in the paper is available online, which is not that common in such domain. Secondly, the dataset has a severe class imbalance, and thirdly the problem of concept drift in casual games is not investigated. Furthermore, the results from the experiments that are carried out on the mobile games dataset pertaining to concept drift are used to evaluate the cross-domain compatibility when these are compared to the results from the iGaming industrial collaborator.

The research paper by Choi et al. (2017) focuses on customer churn prediction in the domain of mobile games. It uses three mobile games, namely Dodge the Mud, TagPro and a racing game with an undisclosed name due to ethical

reasons. Since all three games show similar characteristics all the analysis, experiments and results in this chapter are all linked to the game Dodge the Mud only.

3.8.1 | Dataset Investigation

The dataset of the game Dodge the Mud is available online and is freely downloadable[1]. The raw dataset consists of three columns being the device id, score and time, with the time column being in Unix epoch time format. A sample of the raw dataset is shown in Table 3.1. The dataset spans the date range from January 2015 to May 2016 and contains 153,876 rows, with each device having one or more rows. In order to create a usable dataset for the machine learning model, the initial step is to compute the features and the label. The features are extracted from the score and time column and consist of the active duration, best score and number of times played amongst others, as shall be seen in upcoming sections. Furthermore, to create a label that is used by the supervised machine learning model, churn has to be defined. As with many churn prediction problems the aim of Choi et al. (2017) is to predict whether a new device shall cease gameplay after a predefined number of days from the initial device play date. The definition of churn varies from one business domain to another. In the use case of Choi et al. (2017), a churned customer for casual games is defined as explained next.

Consider a player p having score denoted by s_i at timestamp t_i. Using these definitions, the relative time with reference to the player's first activity timestamp can be computed and defined as t_i^{rel}. With reference to Table 3.1 consider the player p with device id 357470044931974 which has played at least three times with the first play occurring at Unix epoch 1421163783 and the second play happening at Unix epoch 1421164205. Thus t_1^{rel} equates to 0 while t_2^{rel} equates to 422 seconds (1421164205 – 1421163783). Given these definitions, two non-overlapping periods need to be examined. These are the Observation Period (OP) and the Churn Prediction (CP) Period. The OP boundary is used to monitor the player's activity and compute the desired features that are required

[1] `https://zenodo.org/record/166035/files/rawdata_game1.csv?download=1`

device id	score	time
352610060979119	7	1421157320
99000072289368	106	1421163166
352610060979119	0	1421157288
352610060979119	6	1421157344
357470044931974	278	1421163783
357470044931974	420	1421164205
357470044931974	531	1421164262
352217052115825	318	1421206863
354280052161437	15	1421207594

Table 3.1: Dataset sample from the game Dodge the Mud.

by the predictive model, as shall be explained in future sections. The other remaining boundary CP Period is used to label if the player churns or not. OP and CP periods are defined in days. If a player has one or more play log entries in OP and none in CP Period, then that player is tagged as a churner. Otherwise, if the player shows some activity during CP Period, the player is tagged as a non-churner. Figure 3.2 illustrates the aforementioned definitions pictorially where OP is highlighted in blue and CP Period is highlighted in red; both periods are set to two days. Three different examples are shown, two of which are labelled as churn. This can be noted from the lack of activity in the CP Period. More importantly, a player is considered as a churner even though some activity is present after the CP Period. An example of a non-churn player is also shown, and it can be noted that the player has some activity in the CP Period.

OP and CP are parameters in the churn prediction model, and these are highly dependent on the churn prediction problem at hand. In case of the game Dodge the Mud one cannot wait indefinitely to check for player's gameplay, and hence reasonable OP and CP has to be selected. Furthermore, according to Choi et al. (2017), it is evident that the majority of players that churned indeed do not return to play. Hence, it is of utmost importance of selecting reasonable periods. Choi et al. (2017) suggests that having OP of 5 days and CP Period of 10 days offers the best balance between quick decision making and model performance.

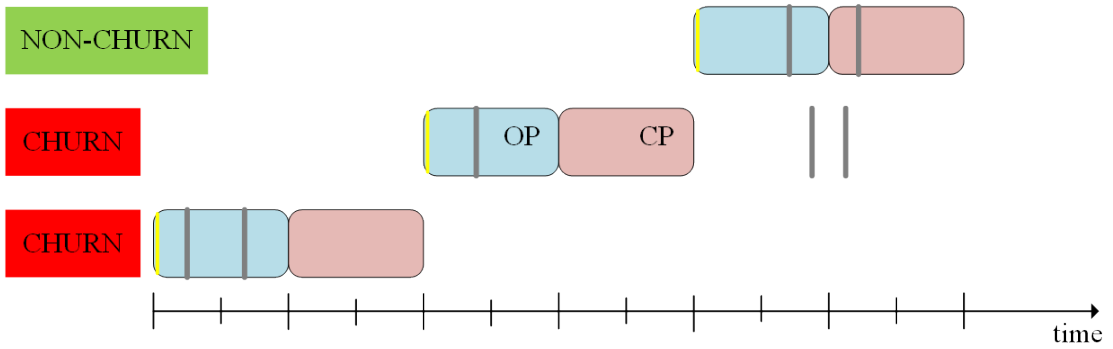


Figure 3.2: Player activity timeline and labelling. Adapted from Choi et al. (2017).

3.8.2 | Replicating the Work of Kim *et* al.

By examining the raw dataset in Table 3.1, one can conclude that the classic supervised machine learning models cannot directly utilise such dataset format since the classification label of churn or non-churn does not exist. Furthermore, each row in the raw dataset only contains a score and its respective timestamp, which renders such format not fit for classification purpose and thus new features need to be created. Moreover, the R code provided was incomplete, and thus dataset creation had to be recreated entirely in order to be able to reproduce the results. The features used and their respective definition and computation are shown in Table 3.2. The OP data available for each unique device was used to generate the features listed. As already mentioned earlier, in order to work out the label (churn/non-churn) the CP and OP periods were used. It was found that around 92% of the total players are churners and thus indicates that the dataset is heavily unbalanced. Furthermore, this imbalance between classes was noticeable in the dataset pertaining to all the three games. In addition to the standard supervised learning dataset (a dataset which contains one or more feature columns and the target column), the work that was carried out by Choi et al. (2017) exhibited another dataset with a different structure used for deep learning; in this case the Long Short Term Memory (LSTM). This dataset was created by slicing the five-day OP period into ten-minute frames, and in each frame, the scores present were summed. This effectively created a vector, each

having 720 elements for each device. The purpose of this dataset was to create an appropriate structure in order to train an LSTM.

Feature	Description	Computation
active_duration	Difference between maximum and minimum play log data or the last vs first play respectively	time of last activity - time of first activity
best_score	Maximum score	max(score)
mean_score	Average score	avg(score)
play_count	Number of times played	count(score)
best_score_index	Best score index normalised by total play count	argmax(score)/ count(score)
best_sub_mean _count	Difference between best and mean score normalised by play count	(max(score)- avg(score))/ count(score)
best_sub_mean _ratio	Difference between best and mean score normalised by the mean score	(max(score)- avg(score))/ mean(score)
consecutive_play _ratio	Count of consecutive play, over total number of adjacent plays. In order to determine whether two adjacent plays qualify as consecutive or not a threshold C exists such that is $c_i < C$	count(c_i)/ (count(score)-1)
sd_score	Standard deviation score	sd(score)
worst_score	Lowest score	min(score)

Table 3.2: List of features extracted per device (player) from the game Dodge the Mud. Reproduced from Choi et al. (2017).

In order to replicate the results obtained by Choi et al. (2017) this dataset was shuffled and split using 80% for training and 20% for testing as this was the approach used by Choi et al. (2017). This dataset was then fitted to several machine learning models mentioned in the paper mainly Logistic Regression (LR), Gradient Boosting Decision Tree (GB), Random Forest (RF) and Long Short Term Memory (LSTM). In order to factor out seed sensitivity, the experiment of splitting and fitting the data to the models mentioned above was repeated several times, and the mean and standard deviation were computed. In the case

of LSTM model, a different dataset was used as mentioned previously in order to create the required input vector. Furthermore, negligible information was given by Choi et al. (2017) about the architecture of the LSTM model. Despite this fact, in order to try and replicate the results of Choi et al. (2017) a 32 node dense layer, a dropout of 0.2 and a sigmoid output layer LSTM model were used. The results obtained for the various machine learning models are shown in Table 3.3. When comparing these results to those obtained by Choi et al. (2017), it can be noted that in case of LR, GB and RF, the results match to approximately 1% to 2% difference. The same cannot be said for the result obtained by the LSTM model as the one obtained by Choi et al. (2017) is roughly 10% better.

Model	Mean	Standard Deviation
Logistic Regression (LR)	0.778	0.030
Gradient Boosting Decision Tree (GB)	0.793	0.012
Random Forest (RF)	0.781	0.015
Long Short Term Memory (LSTM)	0.725	0.005

Table 3.3: Mean and standard deviation for ROC-AUC metric obtained during result replication for the game Dodge the Mud using OP and CP of 5 and 10 days respectively.

3.8.3 | Results and Evaluation

A common practice that is carried out by data science practitioners is evaluating feature importance. In the case of Choi et al. (2017), this was carried out by using the single feature ranking technique. The experiment comprised of correlation testing of each feature with respect to the label column and then by using single feature training and logging the ROC-AUC value. The final feature importance metric was measured by averaging the different results. The outcome for such a ranking technique is shown in Table 3.4. Furthermore, a feature combination experiment was carried out by Choi et al. (2017) in which brute force multiple feature selection was used and the best result achieved for Dodge the Mud happened to be when using all the features excluding *worst_score*. It is important to note that *active_duration* happened to be the most important feature

in both the single feature ranking and brute force multiple feature selection experiments. Furthermore, results show that including more than three features, the improvement in ROC-AUC performance is not significant.

Rank	Feature	ROC-AUC
1	active_duration	0.789
2	play_count	0.768
3	best_sub_mean_ratio	0.743
4	best_score_index	0.732
5	best_score	0.712
6	sd_score	0.691
7	consecutive_play_ratio	0.683
8	best_sub_mean_count	0.672
9	worst_score	0.613
10	mean_score	0.600

Table 3.4: Single feature ranking carried out by Choi et al. (2017) for the game Dodge the Mud. Reproduced from Choi et al. (2017).

The main experiment that the research paper by Choi et al. (2017) highlights is an 80% and 20% split, using the 80% for 10-fold cross-validation and the 20% as the holdout test set. Several machine learning models, are used such as LR, GB, RF, CNN and LSTM. The resulting ROC-AUC for the aforementioned game is shown in Table 3.5. Furthermore, it should be noted that for the main results an OP of 5 days and CP Period of 10 days were used; however, another analysis regarding the effect of OP and CP was carried out, as will be discussed.

Model	ROC-AUC
Logistic Regression (LR)	0.786
Gradient Boosting Decision Tree (GB)	0.791
Random Forest (RF)	0.787
Convolutional Neural Network (CNN)	0.774
Long Short Term Memory (LSTM)	0.792

Table 3.5: ROC-AUC results obtained by Choi et al. (2017) for the game Dodge the Mud using OP and CP of 5 and 10 days respectively. Reproduced from Choi et al. (2017).

Furthermore, other experiments to highlight the importance of the values for OP and CP Period were discussed. Results for different OP and CP Period days show how these two values can impact the model's performance. In particular the larger the OP days, the better the model's predictive performance. The opposite applies for CP Period where the smaller the CP Period, the better the ROC-AUC performance.

3.8.4 | Critique and Limitations

From our review of the paper by Choi et al. (2017), two main shortcomings are noted which we try to address. The first one is related to the fact that the data is not considered as a time series and in the experiments carried out by Choi et al. (2017); the dataset is shuffled. If the dataset is shuffled, the chronological order of events is lost, and different concepts are entwined together, making it impossible to apply concept drift approaches correctly, consequently giving unexpected results. The second is that no detailed investigation on the choice of OP and CP Period is carried out other than arbitrarily testing the performance of the same models under different OP and CP Period. OP and CP periods need to be meticulously chosen via experimentation and have a severe effect on results. Furthermore, in all the results shown, the metric that is used to evaluate performance is Area Under the Receiver Operator Characteristic Curve (ROC-AUC). However, according to Davis and Goadrich (2006) and Rehmsmeier and Saito (2015) the metric Area Under the Precision Recall Curve (PR-AUC) is better suited to cater for heavy class imbalance problems. The studies by Davis and Goadrich (2006) and Rehmsmeier and Saito (2015) highlight how the ROC-AUC curve can be deceptive in gauging classification performance reliability due to the misinterpretation of specificity and how the PR curve gives a more accurate view of the performance because such metric considers true positives amongst predicted positives.

3.9 | Summary

In this chapter, a detailed literature review about churn prediction systems was given. The chapter started by reviewing churn management systems and discussed evident churn drivers that cause customers to defect. Then it was discussed through the use of past literature what feature engineering and selection are employed in churn prediction systems. Moving on to churn prediction approaches be it supervised or unsupervised, and finally, the chapter closes with a discussion of some of the concept drift approaches found in the literature.

4

Methodology

This chapter begins by highlighting some optimisations and implementations on the mobile gaming dataset used in the work of Choi et al. (2017). The chapter then details the design and implementation of a concept drift aware, churn prediction system on the industrial collaborator dataset. As mentioned in previous sections, this work formed part of the collaboration with a local online gaming company, in which the problem of churning customers (which happens to be one of the most common problems in such industry) was tackled. Hence, this chapter explains further the problem at hand and devises the experiments in order to obtain a better solution for predicting churn. In particular, concept drift techniques are explored and their implementation detailed.

4.1 | Addressing the Critique of Kim *et* al.

In order to address the critique discussed in Section 3.8.4, the first part of the work on Kim *et* al. dataset involved reworking the feature set and adding a date feature known as the decision date for each device. This way the dataset is better suited for concept drift experiments. The decision date is equivalent to the end of the OP period and represents the date on which a decision needs to be taken by the machine learning model to predict if a player will churn or not. Furthermore, this work continues by investigating the choice of different OP and CP period and how such values affect the class imbalance of the dataset

amongst other metrics. Additionally, a series of experiments directly related to addressing the problem of concept drift were carried out. First off, an Augmented Dickey Fuller (ADF) statistical test was carried out to check each feature's stationarity, since non-stationarity indicates concept drift. To complement the ADF statistical test, dataset complexity was then analysed by using a Decision Tree (DT). Intuitively one would expect different tree properties when concepts in the data are encountered. Finally, two concept drift approaches were used to determine if there is any difference in performance when compared to the static model. By static model, it is understood that the model is trained only once, and the effects of concept drift are ignored. The first of the concept drift approaches is termed as the Moving Window approach, while the second is known as the Incremental Window approach.

4.1.1 | OP-CP Period Optimisation

The work by Choi et al. (2017) only gives superficial analysis when it comes to choosing OP and CP Period, as it was only mentioned that the machine learning models exhibited improved ROC-AUC performance by increasing OP and decreasing CP Period. However, an analysis of how the underlying structure of the training set is changing was not examined. In order to delve further into such analysis, another period termed the Actual Period (AP) was created. We define AP as a six month period which is placed immediately after CP Period. A timeline highlighting the new period is depicted in Figure 4.1. The main idea behind AP is to check if the device in question shows any activity or not during those six months. By doing so, one can compare the label obtained using CP Period with that obtained in AP. Effectively, each device will have two labels, the predicted label which is governed by the activity between OP and CP Period and the actual label which depends on activity between CP Period and AP. For such analysis, it was deemed that a six month period was adequate rather than waiting indefinitely since such time frame is significantly larger than the OP and CP periods used.

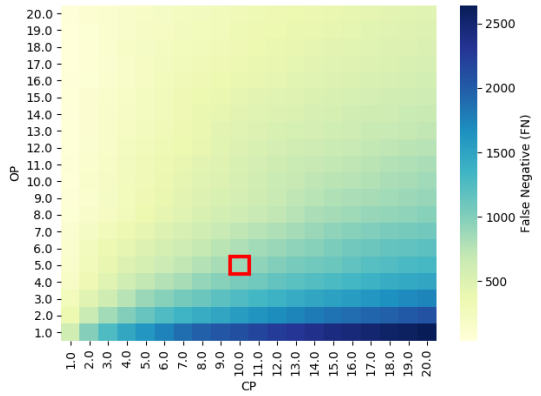
Different OP and CP periods were investigated while keeping AP fixed at six months and each time a confusion matrix was generated. From the confusion

Figure 4.1: Timeline showing the introduced AP period in addition to the known OP and CP periods.

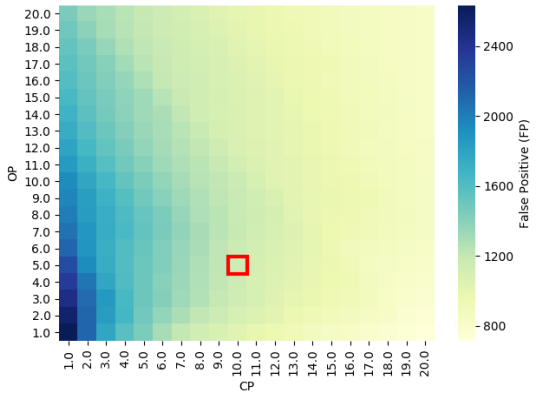
matrix, the recall, precision, accuracy, F1-score and class imbalance were calculated. Figure 4.2 depicts the heat map for various metrics in particular False Negative (FN) (a), False Positive (FP) (b), Recall (c), Precision (d), Class Imbalance (e) and F1 Score (f) for different OP and CP while keeping AP fixed at six months. The reason behind choosing such metrics revolves around the fact that research on customer churn prediction shows that sensitivity is preferred over specificity. Hence, the aim is to primarily reduce the amount of FN and then FP. This is attributed to the fact that incorrectly classifying a churned player, has an associated high cost rather than incorrectly classifying a non-churner (Abou Trab et al., 2015). Choi et al. (2017), used an OP of 5 days and CP of 10 days and this combination is shown as a red square in subfigures of Figure 4.2. It can be noted that by choosing a smaller CP, the amount of FN reduce and consequently even the class imbalance is reduced. Contrastingly, the opposite happens with OP as it can be noted that by increasing OP the amount of FN decreases and thus recall increases.

4.1.2 | Ordered Dataset Experiment

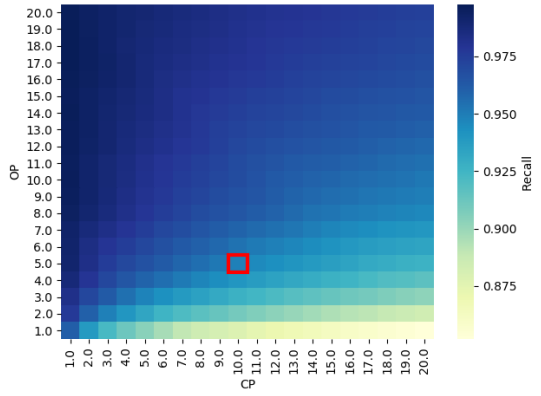
Another experiment that was carried out, which is similar to the main experiment done by Choi et al. (2017) was an 80%, and 20% split on the decision date ordered dataset. Essentially 80% of the data was used to fit several machine learning models, and performance was evaluated on the last 20% holdout set. The results are shown in Table 4.1. Technically, one cannot compare the results shown with those in Table 3.3 as the holdout set differs due to being ordered by decision date. However, this experiment will act as a baseline for the experiments that will follow. Furthermore, the confusion matrix for the Gradient



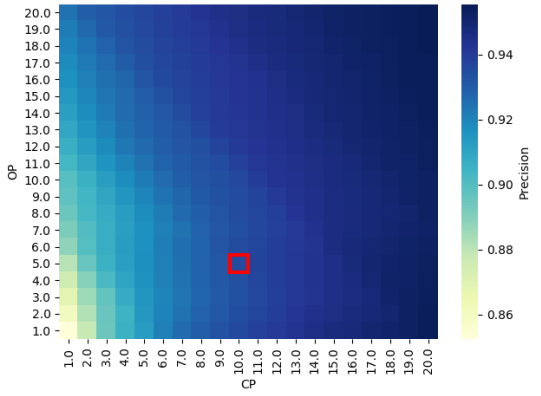
(a) False Negative heat map.



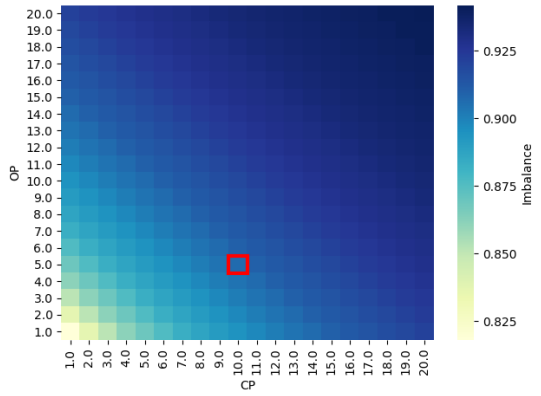
(b) False Positive heat map.



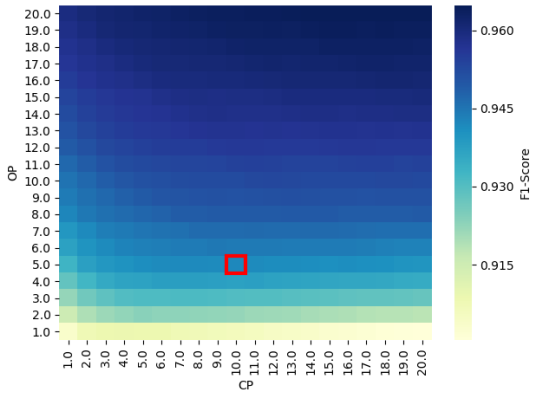
(c) Recall heat map.



(d) Precision heat map.



(e) Class imbalance heat map.



(f) F1 Score heat map.

Figure 4.2: Analysis for different OP and CP at fixed AP of six months.

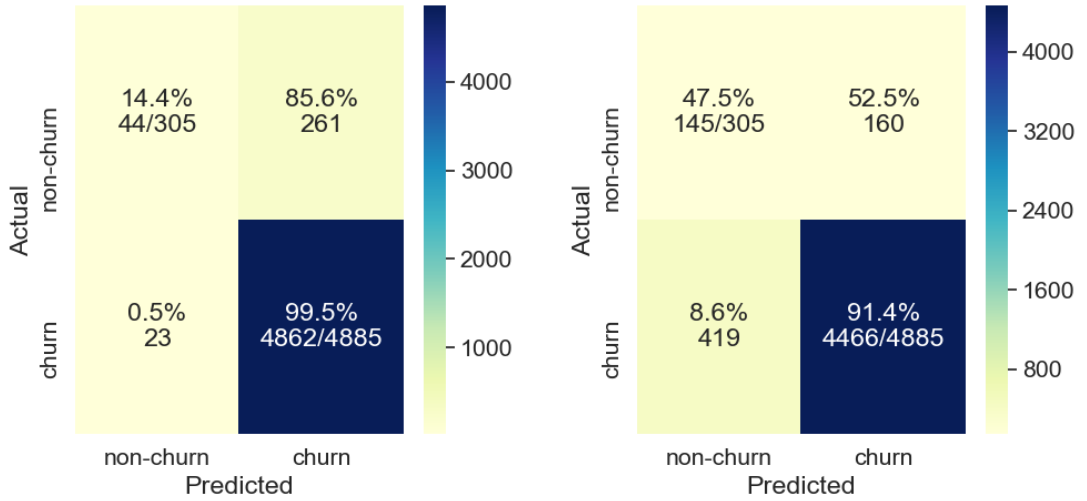
Boosting Decision Tree (GB) model is shown in Figure 4.3a. From the confusion matrix, the high-class imbalance present can be noted and also the fact that the GB model incorrectly classified the majority of the minority class (non-churn) and thus exhibited a high FP rate.

In order to counteract such high-class imbalance, several methods exist such as the Synthetic Minority Oversampling Technique (SMOTE), which was discussed in depth in the background section namely Section 2.5.1. According to Gosain and Sardana (2017), SMOTE outperforms other class imbalance techniques and thus was the technique chosen in this study. The results obtained when implementing the SMOTE technique using several machine learning models on the decision date ordered dataset were tabulated in Table 4.1. It can be noted that there is a marginal difference between the ordered dataset with and without SMOTE. However when investigating the confusion matrix depicted in Figure 4.3b for GB while using SMOTE, the difference is easily noticeable. This is so because despite having a 9% decrease in the churn class prediction, the non-churn churn class prediction was boosted by over 200%.

Model	**Mean**	**Standard Deviation**	**Mean (SMOTE)**	**Standard Deviation (SMOTE)**
Gradient Boosting Decision Tree (GB)	0.786	0.000	0.781	0.001
Random Forest (RF)	0.781	0.001	0.783	0.000
Logistic Regression (LR)	0.780	0.000	0.783	0.003

Table 4.1: Mean and standard deviation for ROC-AUC metric for the game Dodge the Mud using the decision date ordered dataset created and OP and CP of 5 and 10 days respectively; with and without SMOTE class imbalance technique.

Another option of how to mitigate class imbalance is by evaluating the Receiver Operator Characteristic (ROC) curve of the decision date ordered dataset model (Figure 4.4). Effectively the ROC curve describes the model's performance concerning True Positive rate and False Negative rate for different decision thresholds. Thus one can use this metric to compute the best threshold, which strikes



(a) Confusion Matrix for decision date ordered dataset using GB model.

(b) Confusion Matrix for decision date ordered dataset using GB model and SMOTE.

Figure 4.3: Confusion Matrix for decision date ordered dataset using GB model; with and without SMOTE class imbalance technique.

a balance between the two classes. This can be worked out by computing the geometric mean for each threshold used and then finding the index which gives the best mean. In this case, the classification decision threshold resulted in being 0.95, which means that if the predicted probability is greater than 0.95, the outcome is labelled as churn otherwise as non-churn. When using this chosen decision threshold, the confusion matrix resulted in being as depicted in Figure 4.5. Upon investigating the confusion matrix, a classification improvement of the negative class of about 421% can be noticed when compared to the model without SMOTE and 52% increase when compared to the model with SMOTE. Contrastingly a decrease in performance was noted for the positive class having a decrease in performance of about 28% when compared to the model without SMOTE and 21% when compared to the model with SMOTE. This shows how choosing the right decision threshold can help mitigate the problem of high-class imbalance.

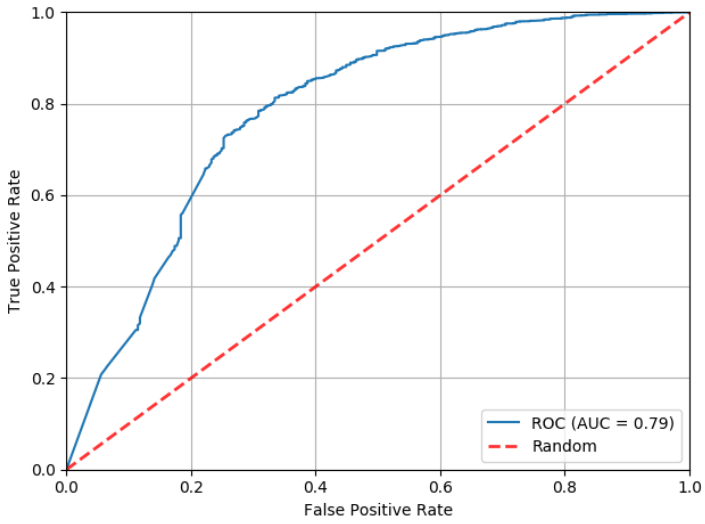


Figure 4.4: ROC curve for GB without SMOTE and using ordered dataset.

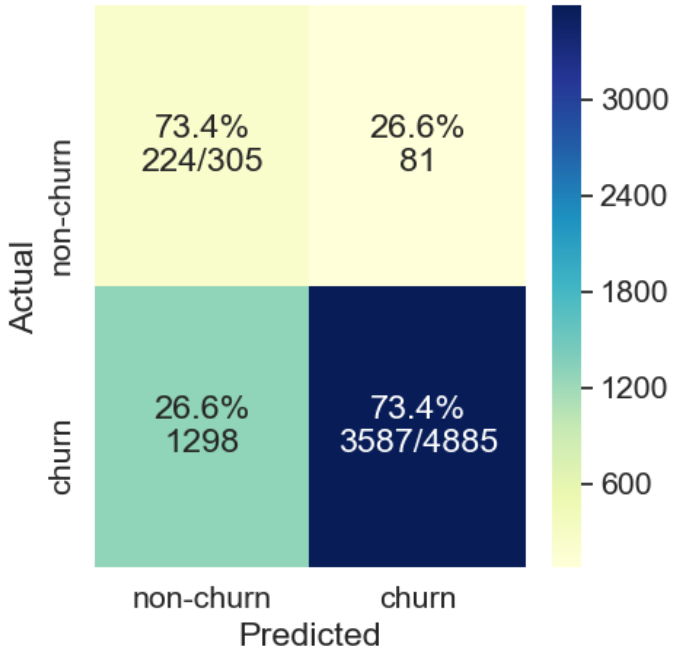


Figure 4.5: Confusion Matrix for decision date ordered dataset using GB model and classification decision threshold of 0.95.

4.1.3 | Concept Drift Investigation

4.1.3.1 | Time Series Analysis

As defined by Žliobaitė (2010), concept drift can be viewed as a learning problem that changes over time. Inherently this portrays concept drift as a non-stationary learning problem that morphs over time. Thus, first of all, stationarity tests need to be carried out on the dataset. One of the tests that are usually used in such scenario is the Augmented Dickey Fuller (ADF) statistical test. A non-stationary time series is one which has a unit root. Having a unit root means that there is no trend in the time series. This stochastic behaviour makes it more difficult for a machine learning model to learn from the data (Herranz, 2017).

In this case, due to the time order, the dataset can be effectively seen as a time series on which the ADF statistical test be applied. The statistical hypotheses for the ADF is defined as:

Hypothesis 0 (Null) *There is a unit root (non-stationary).*

Hypothesis 1 (Alternative) *There is a no unit root (stationary).*

The ADF test was applied to the ordered dataset on the different features, and the resulting p-values were tabulated in Table 4.2. If the resulting p-value is greater than the 0.05 alpha significance threshold, we fail to reject the null hypothesis that a unit root exists and hence indicates concept drift. From the results, it could be noted that several features which are highlighted in bold exhibit stochastic properties and hence there is evidence for concept drift.

4.1.3.2 | Decision Tree Complexity

Given the ADF results obtained, another experiment was devised which makes use of a Decision Tree (DT), to gauge the complexity of the dataset. The motivation behind this experiment using DT revolves around the belief that when encountering concepts in the dataset, the tree properties should also exhibit drifts. In addition to tree properties, one needs to monitor churn properties such as count and ratio in order to learn any relationship that points to concept drift.

Feature	P-value	Outcome
active_duration	0.013	Stationary
best_score	**0.110**	Non-stationary
mean_score	**0.050**	Non-stationary
play_count	0.011	Stationary
best_score_index	0.040	Stationary
best_sub_mean_count	0.022	Stationary
best_sub_mean_ratio	**0.050**	Non-stationary
consecutive_play_ratio	**0.124**	Non-stationary
sd_score	0.027	Stationary
worst_score	**0.051**	Non-stationary

Table 4.2: ADF statistical test p-values on the decision date ordered dataset at a 0.05 level of significance. Bold values highlighting non-stationary features, indicating variability and drift.

In the first DT experiment that was carried out, the dataset was incrementally added to the DT training set. Iteratively, a portion of the dataset was continuously added, and each time a DT was fitted to the data while monitoring the tree depth and node count. The initial window size was that of 30 days and after each run, a single day was added to the original window, and a new DT fitted; hence the name incremental. The result for the incremental window DT complexity is depicted in Figure 4.6. The data included in this experiment was the first 80% of the decision date ordered dataset, and for each iteration, the tree node count, tree depth, churn count, device count and churn ratio were plotted. Effectively, this experiment traverses the first 80% daily and on each day includes all the historical data available, hence the single day increment nomenclature. From Figure 4.6 the similar trend exhibited between the node count, churn count, device count and churn ratio is evident. Additionally, the node count plot does not stabilise and continuously learns new concepts encountered in the ordered dataset. The same cannot be said to the tree depth as it showed a period where the depth was continuously increasing and then stabilises.

Apart from the incremental window experiment which was previously discussed, a moving window DT experiment was carried out. In this experiment, rather than keeping the original dataset and incrementally adding to it, the train-

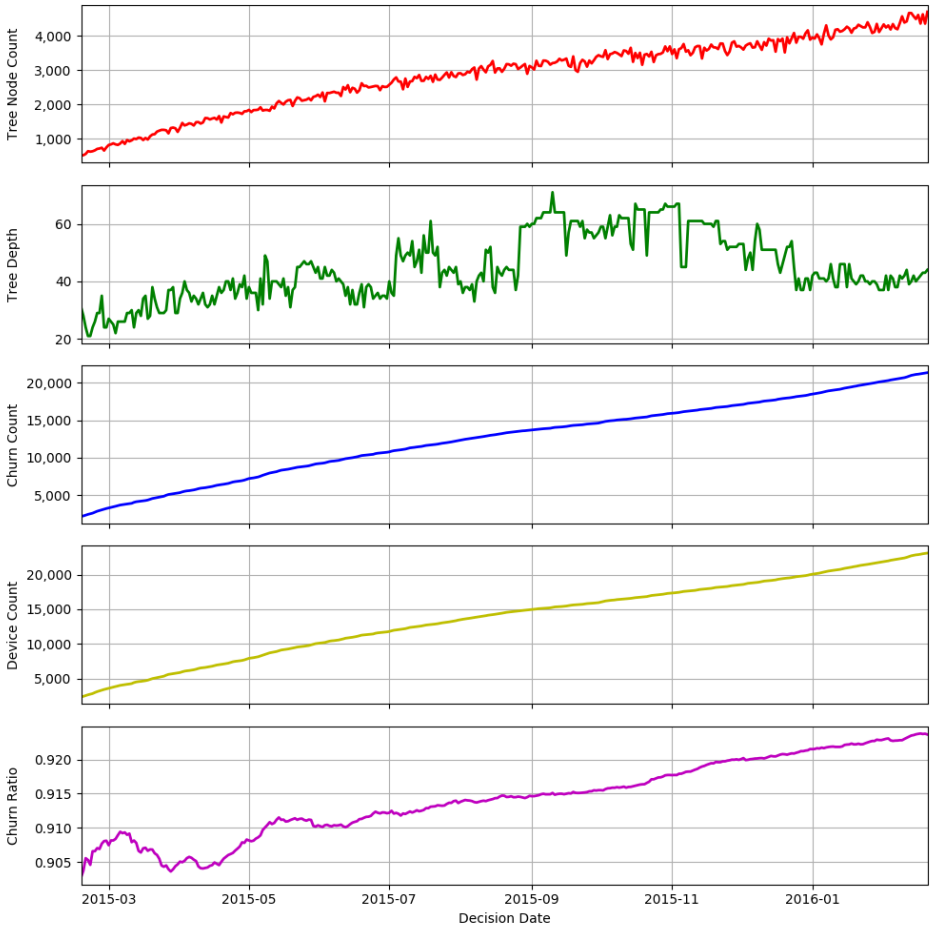


Figure 4.6: Incremental Window DT complexity analysis. Initial window of 30 days with single day increments.

ing set data comprises only of a particular window size which is continuously slid over the 80% portion of the decision date ordered dataset. Thus, in this case, the DT will have the data pertaining to the specific window date range. The result of having a 30-day window and a moving stride (step) of one day is depicted in Figure 4.7. It is once again evident that the node count shows abrupt shifts highlighting the different complexities encountered throughout the dataset.

It might be argued that these shifts can be attributed to the fact that in this test, the training set contains different number of rows because slicing is done on the decision date. It might be the case that a particular window contains less

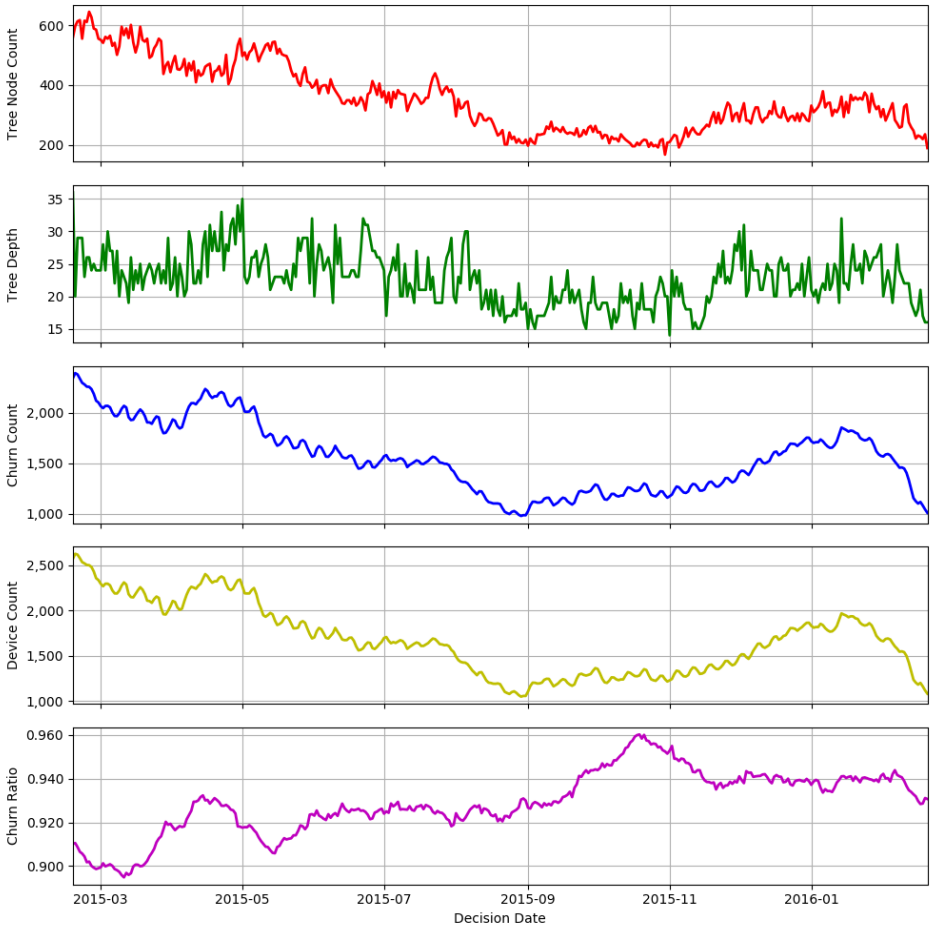


Figure 4.7: Moving Window DT complexity analysis (Window Size = 30 days, Stride = 1 day).

activity. In view of this, another test was carried out where the training set was fixed to the last 2,000 devices, thus creating a fixed-size moving window. The results of such test are illustrated in Figure 4.8. Despite having a fixed training set the shifts in the DT node count is still evident further reinforcing the different concepts present throughout the dataset.

4.1.4 | Concept Drift Mitigation

According to Žliobaitė (2010) under concept drift scenarios, the machine learning model only needs the most recent data to make the best prediction possible.

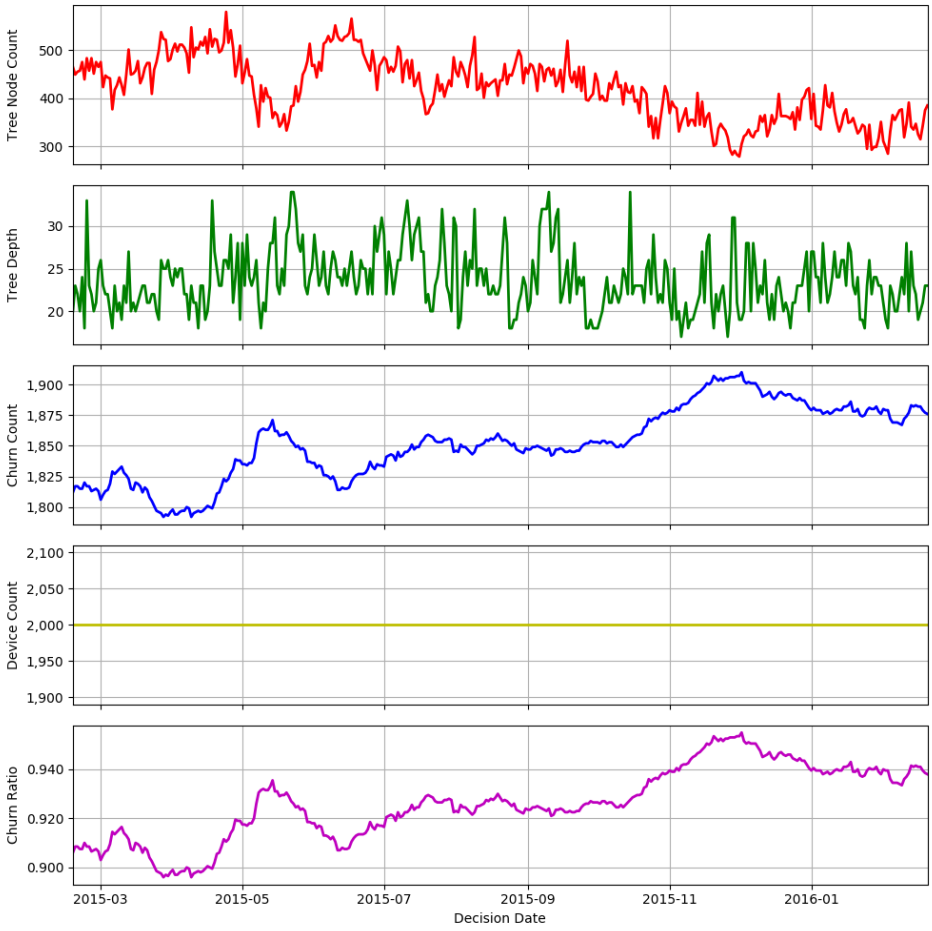


Figure 4.8: Moving Window DT complexity analysis (Window Size = 2,000 devices, Stride = 1 day).

The intuition behind this reasoning is related to the fact that if the model learns the current concepts, it will perform well in future periods until the next learning period arrives. In view of this, this study investigates two concept drift approaches which will be compared to the static model. For the experiments which will be discussed the results are all related to the last 20% of the decision date ordered dataset, as the last 20% portion was used as an evaluation period.

4.1.5 | Moving Window Approach

In this approach, the model only learns from a window of a specific size and predicts an adjacent future period. After a predefined amount of time elapses, the model is retrained on another window and is used to predict future periods. Figure 4.9 gives a high-level overview of what happens when using the Moving Window approach. The orange rectangle labelled 20% represents the last 20% portion of the decision date ordered dataset, which will be used for evaluation purposes. As shown in the illustration, let us consider that the model is trained on a window of 30 days and is used to predict a future window of five days. Thus at a specific point in time, the model learns from the previous 30 day period and predicts the future five day period as shown by window one in Figure 4.9. In this experiment, the dataset is traversed daily and is shown in the figure as a shift of one day. Window two is a replica of window one the only difference being that the whole operation is shifted forward by one day. The same process is repeated until the dataset is exhausted. The results for the moving window approach were compared to the static model which was trained on the first 300 days worth data which approximates roughly to 80% of the dataset. This comparison is crucial in order to find out if implementing the moving window approach achieves better predictive performance. For the evaluation metrics, the F1-Score and PR-AUC were used as they are better suited for high-class imbalance scenarios, as mentioned in Section 3.8.4.

Figure 4.10 illustrates the F1-Score and PR-AUC performance of the GB model when traversing the 20% portion using a moving window size of 30 days and predicting future five days. For comparison, the performance of the static model is also shown. On average, the moving window approach achieved an F1-Score of 0.966 and a PR-AUC of 0.971 while the static model achieved an F1-Score of 0.971 and a PR-AUC of 0.972. It can be deduced that for the game Dodge the Mud, the moving window approach did not show any performance improvement, which can be attributed to the severe class imbalance present.

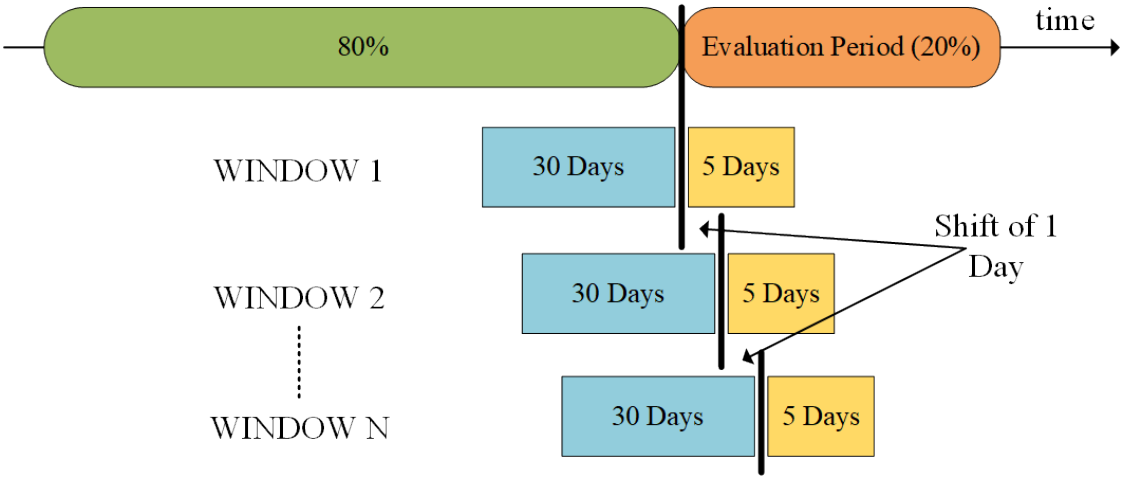


Figure 4.9: Moving Window Approach Overview.

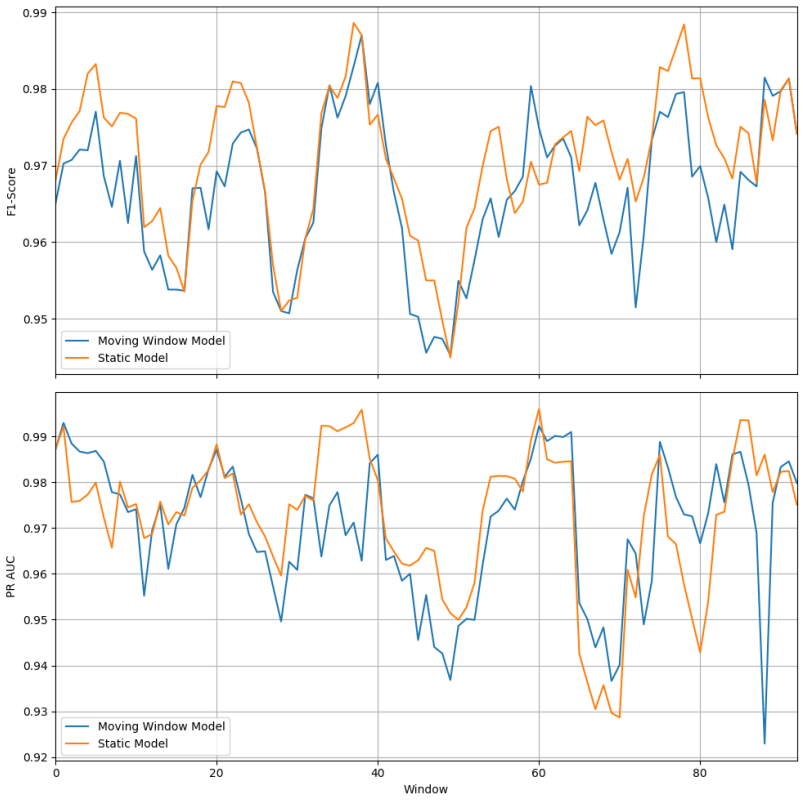


Figure 4.10: Moving Window Approach using GB, window size of 30 days and predicting future 5 days.

4.1.6 | Incremental Window Approach

This approach conceptually is similar to the moving window approach with subtle changes to how the model learns during each iteration. In the incremental window approach, new data is continuously added to the original window after each iteration. Figure 4.11 gives an overview of what happens in the incremental window approach. It can be noted that for window one the model learns on a window size of 30 days. However, in window two, the model will learn on a window size of 31 days and so on until the dataset is exhausted. The amount of data that is added to the initial window size depends on the step that is taken during each iteration. In the case of the figure shown, on each run, there is a shift of one day.

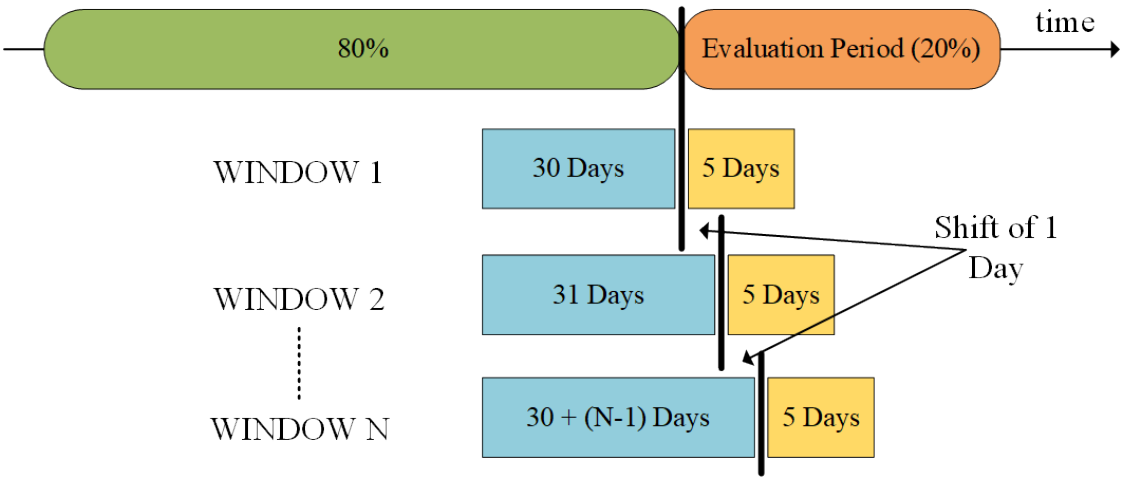


Figure 4.11: Incremental Window Approach Overview.

Figure 4.12 depicts the F1-Score and PR-AUC performance of the GB model when traversing the 20% portion using the incremental window approach with an initial window size of 30 days, predicting future five days. On each iteration, there is a shift of one day. As can be noted, once again, the performance between the static and incremental window model is comparable with instances where one of the models is slightly better performing. However, with regards to average F1-Score and PR-AUC, the incremental window model achieved scores of 0.969 and 0.976, respectively. In contrast, the static model achieved an aver-

age F1-Score of 0.971 while an average PR-AUC of 0.972. These results show a marginal improvement in the PR-AUC metric while a marginal deterioration in F1-Score.

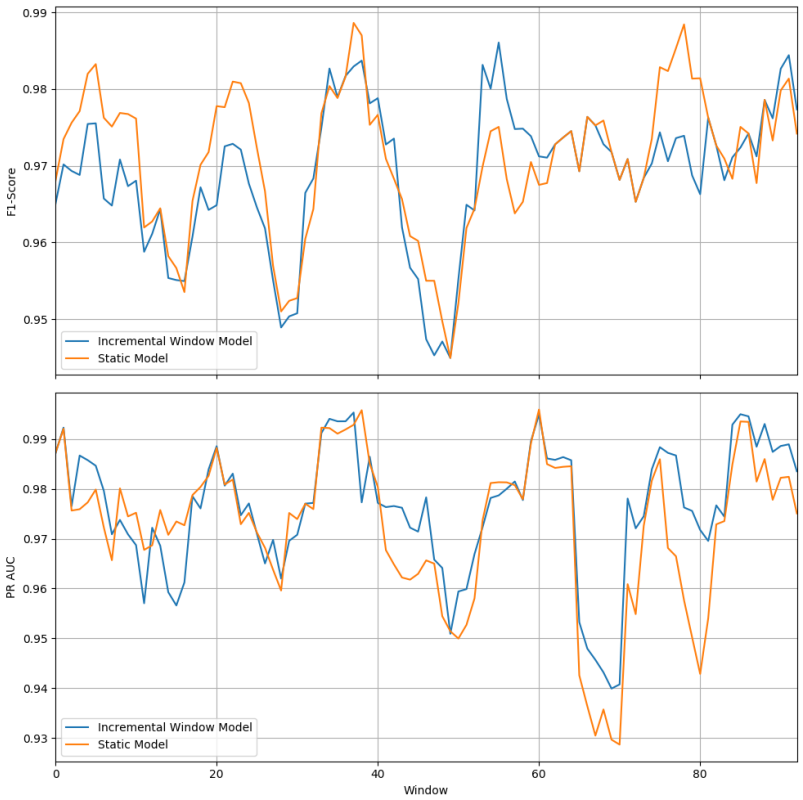


Figure 4.12: Incremental Window Approach using GB, window size of 30 days and predicting future 5 days.

In this section, the work of Choi et al. (2017) was reviewed. Firstly this study replicated the results achieved by the research paper and performed several experiments and analysis on the decision date ordered dataset, which contrasts the research paper, as dataset time order was not taken into consideration. Secondly, an investigation related to the effect of concept drift on the casual mobile game Dodge the Mud was carried out. Thirdly this section discussed two concept drift approaches, namely the moving window approach and the incremental window approach that will also be used on the industrial collaborator dataset. From the results obtained on the game Dodge the Mud, it can be deduced that these ap-

proaches did not show any potential improvement in predictive performance.

4.2 | Collaborator Problem Investigation

In light of the investigation on the work of Choi et al. (2017), this study conducts an investigation whether such concept drift approaches are transferable to other industries namely the iGaming industry. Figure 4.13 depicts the process followed in this study to address the problem of churn prediction in hand. The process starts by extracting the dataset from the industrial collaborator. Feature engineering was then performed on the dataset, which gave rise to other features. Subsequently experiments to identify concept drift were carried out. This part of the work involved the use of statistical tests and also made use of decision trees to analyse dataset complexity. Then, the concept drift framework was designed and implemented. In particular, two approaches are used, namely the Moving Window Approach and the Window Dissimilarity Approach. Lastly, the results were evaluated and compared with those achieved by the industrial collaborator.

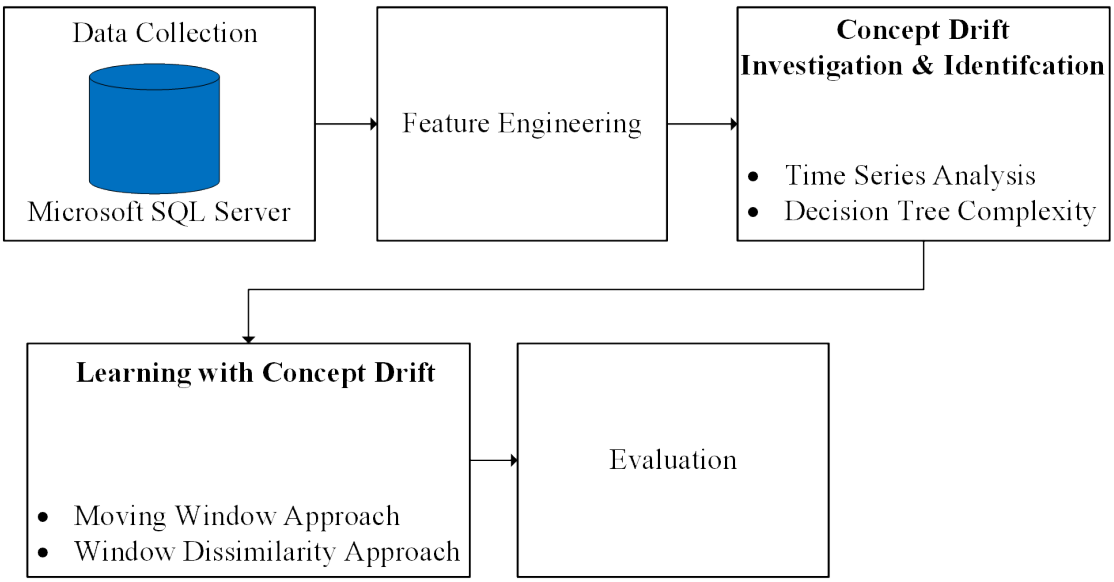


Figure 4.13: High level overview of the methodology steps carried in this work.

Tough competition exists in the local online iGaming industry, hence knowing in advance which customers shall churn, offers a competitive advantage to the company (Grech, 2016; Micallef, 2019). Thus, it is a common practice that such companies have a data science team that has the responsibility of creating predictive models to help tackle such problems. In the case of the industrial collaboration, the data science team has implemented a churn prediction model.

In order to understand the problem of churn, the customer's activities need to be analysed thoroughly. In the case of the industrial partner, customers are acquired daily when they register on the site. At this point, the customers are defined as newly registered customers (NRC). When the respective customers make their first deposit, they are termed as newly depositing customers (NDC). If one tracks the activity of the customers that started depositing on a particular day, the customer journey can be monitored in relation to the churn problem.

For example, suppose that on a particular day x number of customers were tagged as new depositing customers. As time goes by the number of active customers (betting or depositing) starts to reduce or, in other words, churn. This is illustrated in Figure 4.14, which shows how the activity of new depositing customers acquired on a certain day changes as time increases. That is, the plot shows how the activity of several newly depositing customers change with respect to their first deposit date (FDD). According to the industrial collaborator data science team, out of the total number of customers acquired in a particular day, approximately 44% of such customers are retained after seven days; thus having 56% churned customers. This is highlighted by the red dashed line in Figure 4.14.

From the outcome of this plot, it becomes clear that one needs to make a distinction between new registered customers and long term customers. New registered customers refer to those customers that have just registered and thus are in the initial acquisition stage. Long term customers, refer to those customers that have been engaged with the company for an extended period after their first deposit date. According to the industrial collaborator, a long term customer is one that has been engaged with a company for at least 90 days. For this reason, the problem of churn can be split into two main categories being early churn and long term churn. According to the industrial collaborator, the problem of early

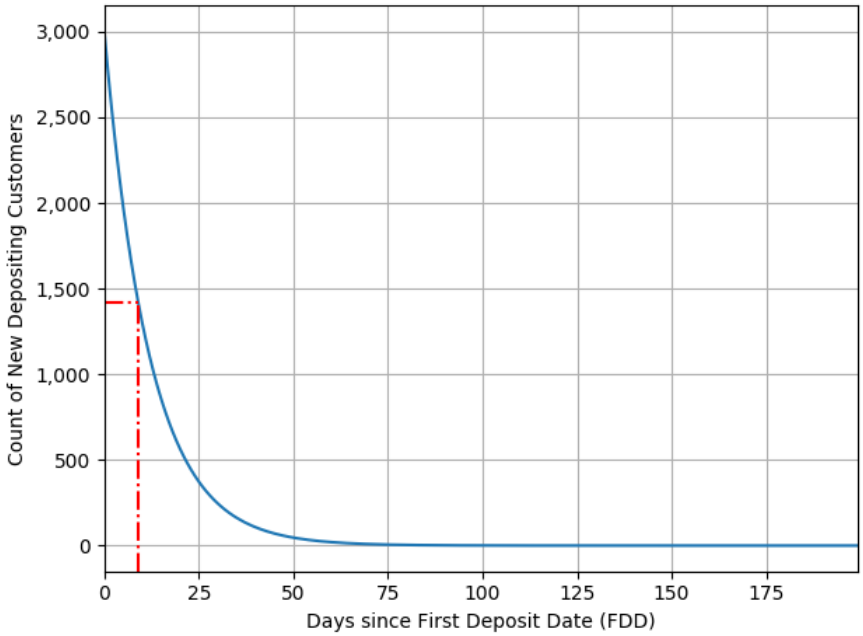


Figure 4.14: Plot showing how the activity of a number of new depositing customers on a particular day decays with time according to the industrial collaborator data.

churn is more important than that of long term churn since the churn rate of new registered customers is higher than that of long term customer. Furthermore, customers who pass the early churn stage tend to be loyal. Loyal customers are willing to continue depositing and being active for a long period, and thus the ratio of churners is smaller. Hence, the focus of this study is on early churn.

Currently, the data science team of the industrial collaborator implemented an early churn prediction model that focuses only on casino customers and is triggered according to the first deposit date of the customer. For the training process, the team used a train-test split procedure and used cross-validation for model selection. The models used were Random Forest (RF) and Light Gradient Boosting Machine (LGBM), with the LGBM outperforming RF by 2.5%. Hence, for this reason, the chosen model for the production environment was chosen to be LGBM. On a daily basis, specific customers who have elapsed five days from their first deposit date are selected, and the churn prediction model is used to

compute the predicted label for each of the selected customers. The prediction results are then sent to both the marketing and Customer Relationship Management (CRM) teams which come up with an ideal strategy on how best to retain the customers. There are various ways of how this can be done, for example, by sending a bonus email to the customer or tailoring a specific campaign depending on the features of the predicted customers, amongst others. One of the most important steps of the process mentioned above is to define the optimal period to trigger a prediction. This is due to the fact that the customer labelling process varies greatly depending on the period chosen, as is seen in Section 4.1.1.

4.2.1 | Early Churn Definition

Many definitions of churn exist, and as discussed in previous sections, they vary according to the domain. The definition of churn, which applies to this study, was discussed with the data science team since their work is used for comparison. Whilst analysing the first few days for each customer during the acquisition stage that is directly after the customer registers, it was noticed that a small portion of the customers made use of their deposited money. This is mainly due to the fact that upon registering a customer might have some welcome bonus which can take several forms. For example, free spins are one of the types, where the customer can play free casino games on the site without using any of his own deposited money.

Through analysis carried by the industrial partner, it resulted that early churn prediction should be made five days after the first deposit date to allow the customers ample time to use up the welcome bonus and start using from the deposited money. The other part of the churn definition focuses on the ideal time frame to label the customers, whether they churned or not. In the case of early churn, a customer is defined as churned if inactivity recurs for a predefined period, five days after the first deposit date. The term inactivity means that the customer did not make any deposit or did not place any bet. With reference to Section 3.8.1, the similarity between the definition of churn between these two different domains is evident as we have the Observation Period (OP) where the customer's activity is analysed and the Churn Prediction (CP) period where the

customer classification label is generated. Figure 4.15 depicts the definition of

Figure 4.15: Early churn definition using the First Deposit Date (FDD) as a reference for all computing the OP and CP periods per customer.

early churn and how the aforementioned OP and CP are used. The first period marked with **blue** which is also referred to as OP is used for feature engineering and the second (CP) marked in **red** is used to create the churn label for the respective customer. According to the data science team, customers who did not show any activity between six and twelve days after the first deposit date, both dates included is labelled as churned. Similarly, the period between the first deposit date and five days within the first deposit date is used to create pertinent features required to make the prediction.

4.3 | Concept Drift Identification

Similar experiments which were introduced during the investigation on the work done by Choi et al. (2017) in particular with respect to concept drift, shall be performed on the dataset of the industrial collaborator. This is done to check the cross-domain compatibility of concept drift mitigation techniques. A number of experiments were devised on the industrial collaborator dataset to determine how the underlying distribution of each feature changes over time. During model training, the industrial collaborator ignored the time ordering of the dataset, which, as is discussed in Chapter 3.8 has an impact on model performance under concept drift circumstances.

4.3.1 | Time Series Analysis

Firstly, experiments were carried out in order to investigate if concept drift exists in the industrial collaborator dataset. Concept drift is linked to data stationarity, which refers to how a time series retains its statistical properties (for example mean, covariance and variance) over time. Stationarity does not imply that the time series values do not change, but that the changes do not alter its distributional properties. Two tests which are typically used in testing stationarity are the Augmented Dickey Fuller (ADF) and Kwiatkowski Phillips Schmidt Shin (KPSS) tests (Herranz, 2017; Zuo, 2019). The former is used to discover the possibility of a unit root in the series. The null hypothesis for this test is that the series has a unit root.

Conversely, the alternative hypothesis is that the series does not have a unit root. Consequently, if the null hypothesis fails to be rejected suggests a non-stationary series. Contrastingly, the latter, while also being a unit root test, has an opposite hypothesis definition and also is non-parametric. Hence, the null hypothesis indicates a trend stationary time series while the alternate points to having a unit root (Balakrishnan et al., 2004). These two tests are typically executed in tandem to ensure true stationarity. However, if any of the tests indicate a unit root, there is significant evidence that the underlying distribution exhibits a changing trend or concept. All the features in the dataset were tested using both tests, as mentioned earlier for each first deposit date in the dataset. Categorical variables could be computed using the above tests because they were transformed by summing up the values on that day. One might argue on the effectiveness of these tests for categorical variables because they lack the definition of mean and variance. However, these were included for completeness (Herranz, 2017; Zuo, 2019).

4.3.2 | Decision Tree Complexity Experiment

After analysing dataset stationarity which indicates whether the dataset suffers from concept drift, another experiment was devised to investigate concept drift further. According to a study carried by Al-Otaibi et al. (2015), it was found

that decision trees are versatile machine learning models that can be adapted and used in many different scenarios. Because of this, we decided to adopt the DT to be used for concept drift analysis rather than for the typical supervised machine learning scenario. This experiment aims to exploit the decision tree in order to discern concepts in the data, with the intuition that if a decision tree is not constrained with any limits to properties such as tree depth and tree node count, concept drift can be highlighted as abnormalities in these properties. Thus this experiment was done by observing decision tree properties such as the tree depth and node count while traversing the first deposit date ordered dataset. Whilst doing so, other dataset properties such as churn count, customer count and churn ratio were noted.

Consider the scenario where the industrial collaborator was ordered by the first deposit date, hence having a minimum and a maximum date respectively. Starting from the minimum date, all the customers available for that date were selected, and a decision tree was fitted. Subsequently, the aforementioned parameters were observed and noted. This step was repeated until the dataset was exhausted. It should be noted that the minimum date present was 2018-01-01, and the maximum was 2020-04-30. Figure 4.16 illustrates an overview of this experiment where a particular first deposit date t is shown in red.

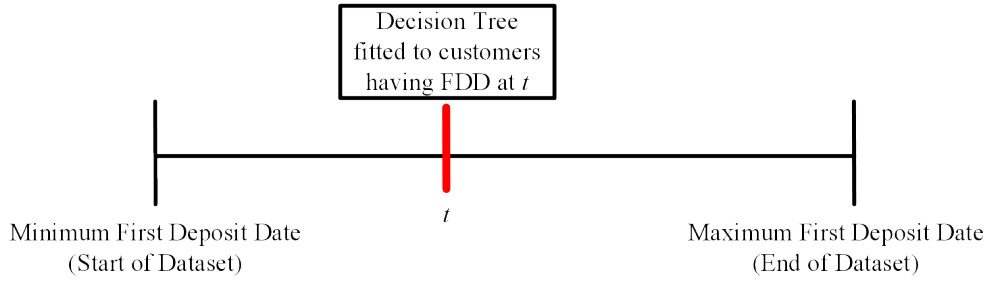


Figure 4.16: Overview of the Decision Tree complexity experiment.

4.3.3 | Decision Tree Feature Importance Experiment

In continuation with the decision tree complexity experiment, it was argued that if the decision tree depth and node count exhibit abnormalities when encoun-

tering different trends or concepts in the data, these tend to be also reflected in other properties of the dataset; in particular the feature importance. That is if one monitors how the feature importance for selected features change with time, there should be trends in the data relating to these concepts. Hence another experiment was devised similar to that discussed in Section 4.3.2 with the difference that in this case, the feature importance is monitored in relation to the churn ratio. For every first deposit date present in the dataset, a decision tree was fitted, and the feature importance for each feature was observed. This experiment aims to reinforce further the occurrence of concept drift throughout the first deposit date ordered dataset.

4.4 | Learning with Concept Drift

According to Liu et al. (2018), if concept drift is not dealt with, the predictive performance decreases. Figure 4.17 illustrates a generic framework for dealing with concept drift. It can be noted that on each iteration, new data is tested for concept drift. If detected, a different approach is taken depending on which concept drift procedure is chosen. In view of Figure 4.17, consider the whole stream of data (including past data) as D. In this work, D is split up into different windows ranging from 1,2..,n where n represents the total number of windows. Each window has a corresponding time t, in this case, related to the first deposit date of the customer. In the production system, data is continually added to the stream and hence creating new windows. Each window has a fixed number of rows, and thus, one needs to chose the window size carefully. Research shows that a small window gives a better view of the current concepts and distribution; however, it may lack the volume of data to properly train and update the model (Liu et al., 2018). In this study, two approaches were devised in order to gauge the model's predictive performance under concept drift scenarios which are the Moving Window Approach and the Window Dissimilarity Approach.

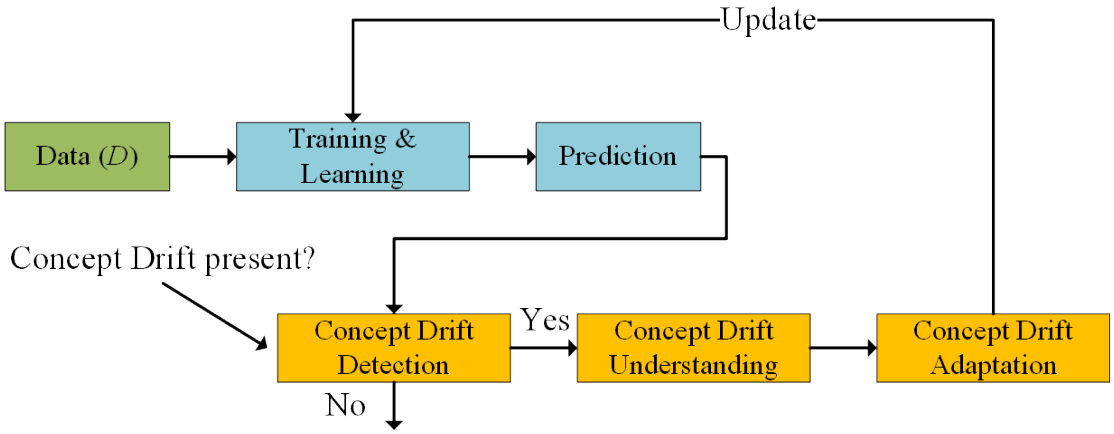


Figure 4.17: Concept Drift Framework. Adapted from Liu et al. (2018).

4.4.1 | Moving Window Approach

In this approach, every new window encountered a new model is retrained on the new data and does not include any other data from past windows. This new model is then used to predict an adjacent future test window. Figure 4.18 highlights the mechanism behind the moving window approach, where the last 25% of the first deposit date ordered dataset was shown as the evaluation period.

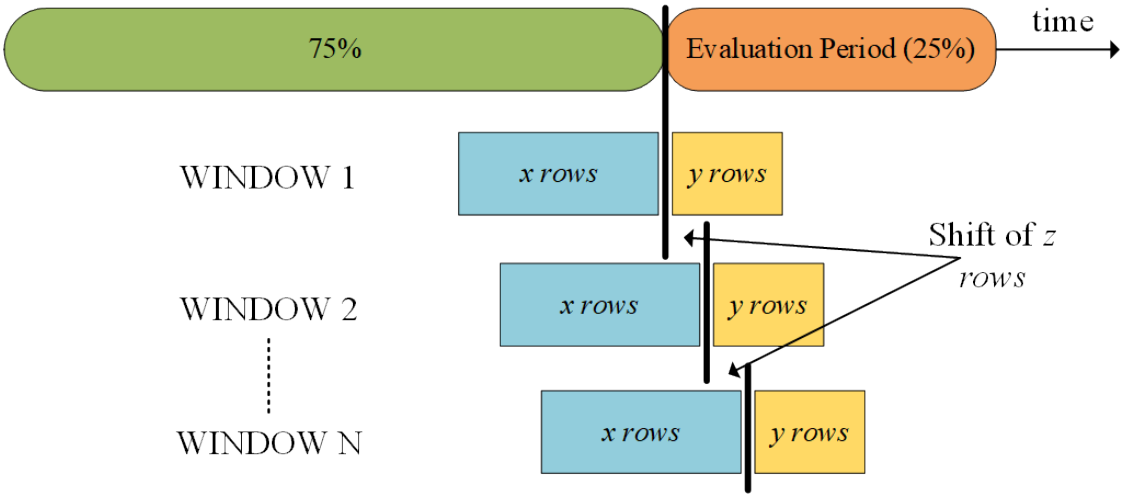


Figure 4.18: Overview of the Moving Window Approach.

Consider that one is at the start of the last 25% portion of the dataset. Let us assume that the machine learning model is fitted to x number of rows (before the start of the 25%) as shown in Window 1 of Figure 4.18. It is pertinent to note that a row in the dataset represents a customer having a particular first deposit date; thus, customer and row are used interchangeably. Once trained, this model is used to predict y number of future rows, and the performance results are stored. This effectively creates one iteration of the whole process. For the next iteration, the window is shifted by z rows forward in the 25% portion as illustrated in the figure. A new machine learning model is trained on x rows and used to predict the future y rows. The process is repeated until the whole 25% portion of the dataset is exhausted. Different training and testing sizes for x and y respectively were tried. For the value of x 50,000, 100,000 and 150,000 were used while for y 1,000 and 5,000 rows were used. The dataset was traversed in 1,000 row increments (refer to z in the figure).

In addition to the moving window model, this approach used another model termed the static model which shadows the approach that is used by the industrial collaborator; that is fit the model to a portion of the dataset and ignore the effects of concept drift. Hence, several experiments were carried out where the static model was fitted to specific portions of the ordered dataset, namely the first 20%, first 50% and first 75%. This effectively results in three different static models. For each of these three static models, a full run of the moving window approach described is carried each time varying the x and y parameters. During each iteration, the moving window model was fitted to x rows and predicts y rows. The static model was also used to predict the same y rows. This was done to evaluate the performance of the moving window scheme versus the static model, which is only trained once.

4.4.2 | Window Dissimilarity Approach

Another approach that was used in this study was termed as the Window Dissimilarity approach. Conceptually, it is similar to the Moving Window Approach; however, it differs from the aspect if a model fitting is required or not as shall be discussed. Figure 4.19 shows the operation of the Window Dissimilarity

Approach. Similar to the Moving Window Approach in the first window (Window 1 in the figure) the model is fitted to x rows and is used to predict future y rows. This new model and the dataset properties on which it was trained are observed. This is shown highlighted by the violet rectangle in the figure. However when it comes to the second iteration (Window 2) after the shift of z rows rather than blindly fitting a new model on the x rows of Window 2, the dissimilarity between the dataset on which the current model (violet rectangle) was fitted on, and the dataset of the current window (Window 2) is evaluated.

The intuition behind this approach is that if the underlying distribution of the dataset is different, there is evidence of concept drift which according to Liu et al. (2018) prediction performance degrades if the model is not updated. By measuring dataset dissimilarity, we effectively measure the severity of the drift, and in order to do so, statistical tests need to be used. The statistical test that was chosen to determine dataset dissimilarity was the Kolmogorov-Smirnov (KS) test. Suppose the outcome of this statistical test suggests there is a dissimilarity. In that case, the current machine learning is updated by training a new model using only the data in the current window. Intuitively, by training a new model which is closer to the data concept, the better predictive performance is achieved. This process is repeated until the 25% portion is exhausted. Similarly to the Moving Window Approach, the static model is used to compare the performance. One benefit of such an approach is that the number of times to fit a model can be reduced if the statistical test deems the current window as similar to that used by the current machine learning model.

4.5 | Implementation

In light of the previous analysis, the implementation section shall be discussed. In particular, an overview of the dataset shall be given and how the concept drift approaches were implemented.

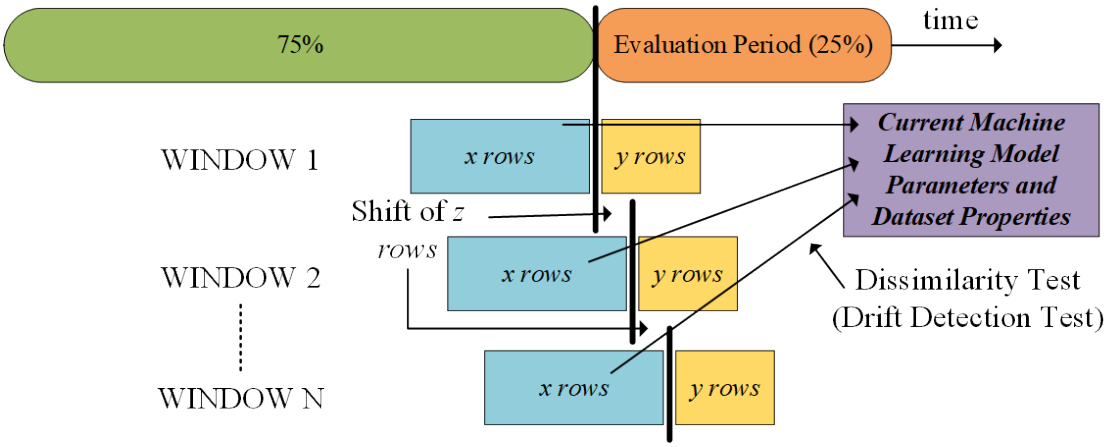


Figure 4.19: Overview of the Window Dissimilarity Approach.

4.5.1 | Dataset

The dataset used in this work was extracted in collaboration with the data science team of the industrial partner, given that the team had already implemented a churn model and thus had already prepared a dataset for modelling purposes. Due to privacy issues, the dataset was anonymised appropriately in order to ensure that it shall be impossible to discern a specific customer. Through various studies, researchers showed that churn prediction datasets are heavily based on the so-called Recency, Frequency and Monetary (RFM) feature structure which was discussed in Section 3 (Alhaery and Suh, 2016; Coussement and De Bock, 2013).

Table 4.3 shows in tabular format, the list of variables used in this study and also includes a short description of the respective feature. As can be seen, there are 149 features related to a customer, and that is not the full list of features as feature engineering produced other features as shall be discussed later on. The dataset used in this work considered transactional data spanning two and a half years, from January 2018 till April 2020. During this span, around 1.4 million customers were recorded, of which around 56% were churned customers. This indicates a slight class imbalance.

The variables shown are part of the raw data that was extracted. For example, the term *Url Referrer* refers to a URL that contains information on how

the customer was referred to the casino be it from google advertisements and Facebook, amongst others. The *Acquisition* field discerns from which channel the customer was acquired be it from an affiliate channel, direct channel or organic channel. The *Same Day Conversion* field indicates whether the customer's account verification date also occurred on the first deposit date. Then there are a plethora of fields related to betting and winning activity which are divided according to the game category and operating platform. Game category refers to the different casino game types available such as poker, slots, blackjack and so on.

In contrast, the operating platform refers to where these bets or wins originated bet it from a desktop computer or mobile. Furthermore betting and winning amount are categorised into three types of funds being the real, locked and bonus such as the *Bet Real Mobile*, *Bet Locked Mobile* and *Bet Bonus Mobile* shown. The real category refers to actual withdrawable money. The locked variant refers to money that cannot be withdrawn until a wagering requirement is met and the bonus variant refers to money associated with in-game bonuses. Furthermore, there are several fields related to deposits and withdrawals and are categorised depending from where the transaction originated be it from a Payment Service Provider (PSP) or a bank card. For the different types of fields aforementioned, both the count and the amount were included.

Name	Type	Description
Url Referrer	Categorical	How was the customer referred to the casino site.
Acquisition	Categorical	How was the customer acquired.
Gender	Categorical	customer gender.
Age	Numeric	Age of the customer.
Verification Date	Datetime	Date and time of customer verification.
Same Day Conversion	Boolean	If first deposit and account verification fall on the same day.
Country	Categorical	Country of registration.
Bet Mobile	Numeric	Amount of bets coming from a mobile platform.
Bet Desktop	Numeric	Amount of bets coming from a desktop platform.
Win Mobile	Numeric	Amount of wins coming from a mobile platform.
Win Desktop	Numeric	Amount of wins coming from a desktop platform.
Jackpot Contribution Mobile	Numeric	Amount of jackpot contribution coming from a mobile platform.
Jackpot Contribution Desktop	Numeric	Amount of jackpot contribution coming from a desktop platform.

Continued on next page

Table 4.3 – continued from previous page

Name	Type	Description
Number of Rounds Mobile	Numeric	Bet count coming from a mobile platform.
Number of Rounds Desktop	Numeric	Bet count coming from a desktop platform.
Bet Bonus Mobile	Numeric	Amount of bonus bets coming from a mobile platform.
Bet Bonus Desktop	Numeric	Amount of bonus bets coming from a desktop platform.
Bet Locked Mobile	Numeric	Locked wallet amount attributed to bets on a mobile platform.
Bet Locked Desktop	Numeric	Locked wallet amount attributed to bets on a desktop platform.
Bet Real Mobile	Numeric	Real wallet amount attributed to bets on a mobile platform.
Bet Real Desktop	Numeric	Real wallet amount attributed to bets on a desktop platform.
Win Bonus Mobile	Numeric	Amount of bonus wins coming from a mobile platform.
Win Bonus Desktop	Numeric	Amount of bonus wins coming from a desktop platform.
Win Locked Mobile	Numeric	Locked wallet amount attributed to wins on a mobile platform.
Win Locked Desktop	Numeric	Locked wallet amount attributed to wins on a desktop platform.
Win Real Mobile	Numeric	Real wallet amount attributed to wins on a mobile platform.
Win Real Desktop	Numeric	Real wallet amount attributed to wins on a desktop platform.
Jackpot Contribution	Numeric	Amount of jackpot contribution platform independent.
Jackpot Win	Numeric	Amount of jackpot win platform independent.
Number of Rounds Slots	Numeric	Bet count coming from slots.
Number of Rounds Blackjack	Numeric	Bet count coming from blackjack.
Number of Rounds Table Game	Numeric	Bet count coming from table games.
Number of Rounds Roulette	Numeric	Bet count coming from roulette games.
Number of Rounds Lottery	Numeric	Bet count coming from lottery games.
Number of Rounds Poker	Numeric	Bet count coming from poker games.
Number of Rounds Bingo & Keno	Numeric	Bet count coming from bingo and keno games.
Number of Rounds Live Casino	Numeric	Bet count coming from live casino games.
Number of Rounds Jackpot	Numeric	Bet count coming from jackpot games.
Number of Rounds Other	Numeric	Bet count coming from neither of the previous games.
Bet Slots	Numeric	Bet amount coming from slots.
Bet Blackjack	Numeric	Bet amount coming from blackjack.

Continued on next page

Table 4.3 – continued from previous page

Name	Type	Description
Bet Table Game	Numeric	Bet amount coming from table games.
Bet Roulette	Numeric	Bet amount coming from roulette games.
Bet Lottery	Numeric	Bet amount coming from lottery games.
Bet Poker	Numeric	Bet amount coming from poker games.
Bet Bingo & Keno	Numeric	Bet amount coming from bingo and keno games.
Bet Live Casino	Numeric	Bet amount coming from live casino games.
Bet Jackpot	Numeric	Bet amount coming from jackpot games.
Bet Other	Numeric	Bet amount coming from neither of the previous games.
Win Slots	Numeric	Win amount coming from slots.
Win Blackjack	Numeric	Win amount coming from blackjack.
Win Table Game	Numeric	Win amount coming from table games.
Win Roulette	Numeric	Win amount coming from roulette games.
Win Lottery	Numeric	Win amount coming from lottery games.
Win Poker	Numeric	Win amount coming from poker games.
Win Bingo & Keno	Numeric	Win amount coming from bingo and keno games.
Win Live Casino	Numeric	Win amount coming from live casino games.
Win Jackpot	Numeric	Win amount coming from jackpot games.
Win Other	Numeric	Win amount coming from neither of the previous games.
Bet Real Slots	Numeric	Bet real amount coming from slots.
Bet Real Blackjack	Numeric	Bet real amount coming from blackjack.
Bet Real Table Game	Numeric	Bet real amount coming from table games.
Bet Real Roulette	Numeric	Bet real amount coming from roulette games.
Bet Real Lottery	Numeric	Bet real amount coming from lottery games.
Bet Real Poker	Numeric	Bet real amount coming from poker games.
Bet Real Bingo & Keno	Numeric	Bet real amount coming from bingo and keno games.
Bet Real Live Casino	Numeric	Bet real amount coming from live casino games.
Bet Real Jackpot	Numeric	Bet real amount coming from jackpot games.
Bet Real Other	Numeric	Bet real amount coming from neither of the previous games.
Bet Bonus Slots	Numeric	Bet bonus amount coming from slots.
Bet Bonus Blackjack	Numeric	Bet bonus amount coming from blackjack.
Bet Bonus Table Game	Numeric	Bet bonus amount coming from table games.
Bet Bonus Roulette	Numeric	Bet bonus amount coming from roulette games.
Bet Bonus Lottery	Numeric	Bet bonus amount coming from lottery games.
Bet Bonus Poker	Numeric	Bet bonus amount coming from poker games.
Bet Bonus Bingo & Keno	Numeric	Bet bonus amount coming from bingo and keno games.
Bet Bonus Live Casino	Numeric	Bet bonus amount coming from live casino games.
Bet Bonus Jackpot	Numeric	Bet bonus amount coming from jackpot games.
Bet Bonus Other	Numeric	Bet bonus amount coming from neither of the previous games.
Bet Locked Slots	Numeric	Bet locked amount coming from slots.
Bet Locked Blackjack	Numeric	Bet locked amount coming from blackjack.
Bet Locked Table Game	Numeric	Bet locked amount coming from table games.
Bet Locked Roulette	Numeric	Bet locked amount coming from roulette games.
Bet Locked Lottery	Numeric	Bet locked amount coming from lottery games.

Continued on next page

Table 4.3 – continued from previous page

Name	Type	Description
Bet Locked Poker	Numeric	Bet locked amount coming from poker games.
Bet Locked Bingo & Keno	Numeric	Bet locked amount coming from bingo and keno games.
Bet Locked Live Casino	Numeric	Bet locked amount coming from live casino games.
Bet Locked Jackpot	Numeric	Bet locked amount coming from jackpot games.
Bet Locked Other	Numeric	Bet locked amount coming from neither of the previous games.
Win Real Slots	Numeric	Win real amount coming from slots.
Win Real Blackjack	Numeric	Win real amount coming from blackjack.
Win Real Table Game	Numeric	Win real amount coming from table games.
Win Real Roulette	Numeric	Win real amount coming from roulette games.
Win Real Lottery	Numeric	Win real amount coming from lottery games.
Win Real Poker	Numeric	Win real amount coming from poker games.
Win Real Bingo & Keno	Numeric	Win real amount coming from bingo and keno games.
Win Real Live Casino	Numeric	Win real amount coming from live casino games.
Win Real Jackpot	Numeric	Win real amount coming from jackpot games.
Win Real Other	Numeric	Win real amount coming from neither of the previous games.
Win Bonus Slots	Numeric	Win bonus amount coming from slots.
Win Bonus Blackjack	Numeric	Win bonus amount coming from blackjack.
Win Bonus Table Game	Numeric	Win bonus amount coming from table games.
Win Bonus Roulette	Numeric	Win bonus amount coming from roulette games.
Win Bonus Lottery	Numeric	Win bonus amount coming from lottery games.
Win Bonus Poker	Numeric	Win bonus amount coming from poker games.
Win Bonus Bingo & Keno	Numeric	Win bonus amount coming from bingo and keno games.
Win Bonus Live Casino	Numeric	Win bonus amount coming from live casino games.
Win Bonus Jackpot	Numeric	Win bonus amount coming from jackpot games.
Win Bonus Other	Numeric	Win bonus amount coming from neither of the previous games.
Win Locked Slots	Numeric	Win locked amount coming from slots.
Win Locked Blackjack	Numeric	Win locked amount coming from blackjack.
Win Locked Table Game	Numeric	Win locked amount coming from table games.
Win Locked Roulette	Numeric	Win locked amount coming from roulette games.
Win Locked Lottery	Numeric	Win locked amount coming from lottery games.
Win Locked Poker	Numeric	Win locked amount coming from poker games.
Win Locked Bingo & Keno	Numeric	Win locked amount coming from bingo and keno games.
Win Locked Live Casino	Numeric	Win locked amount coming from live casino games.
Win Locked Jackpot	Numeric	Win locked amount coming from jackpot games.
Win Locked Other	Numeric	Win locked amount coming from neither of the previous games.
Sum Deposits PSP	Numeric	Amount of deposits coming from a PSP.

Continued on next page

Table 4.3 – continued from previous page

Name	Type	Description
Sum Deposits Bank Card	Numeric	Amount of deposits not coming from a PSP.
Sum Withdrawals PSP	Numeric	Amount of withdrawals coming from a PSP.
Sum Withdrawals Bank Card	Numeric	Amount of withdrawals not coming from a PSP.
Sum Failed Deposits PSP	Numeric	Amount of failed deposits coming from a PSP.
Sum Failed Deposits Bank Card	Numeric	Amount of failed deposits not coming from a PSP.
Sum Failed Withdrawals PSP	Numeric	Amount of failed withdrawals coming from a PSP.
Sum Failed Withdrawals Bank Card	Numeric	Amount of failed withdrawals not coming from a PSP.
Sum Cancelled Withdrawals PSP	Numeric	Amount of cancelled withdrawals coming from a PSP.
Sum Cancelled Withdrawals Bank Card	Numeric	Amount of cancelled withdrawals not coming from a PSP.
Sum Reversed Withdrawals PSP	Numeric	Amount of reversed withdrawals coming from a PSP.
Sum Reversed Withdrawals Bank Card	Numeric	Amount of reversed withdrawals not coming from a PSP.
Sum Rejected Withdrawals PSP	Numeric	Amount of rejected withdrawals coming from a PSP.
Sum Rejected Withdrawals Bank Card	Numeric	Amount of rejected withdrawals not coming from a PSP.
Count Deposits PSP	Numeric	Number of deposits coming from a PSP.
Count Deposits Bank Card	Numeric	Number of deposits not coming from a PSP.
Count Withdrawals PSP	Numeric	Number of withdrawals coming from a PSP.
Count Withdrawals Bank Card	Numeric	Number of withdrawals not coming from a PSP.
Count Failed Deposits PSP	Numeric	Number of failed deposits coming from a PSP.
Count Failed Deposits Bank Card	Numeric	Number of failed deposits not coming from a PSP.
Count Failed Withdrawals PSP	Numeric	Number of failed withdrawals coming from a PSP.
Count Failed Withdrawals Bank Card	Numeric	Number of failed withdrawals not coming from a PSP.
Count Cancelled Withdrawals PSP	Numeric	Number of cancelled withdrawals coming from a PSP.
Count Cancelled Withdrawals Bank Card	Numeric	Number of cancelled withdrawals not coming from a PSP.

Continued on next page

Table 4.3 – continued from previous page

Name	Type	Description
Count Reversed Withdrawals PSP	Numeric	Number of reversed withdrawals coming from a PSP.
Count Reversed Withdrawals Bank Card	Numeric	Number of reversed withdrawals not coming from a PSP.
Count Rejected Withdrawals PSP	Numeric	Number of rejected withdrawals coming from a PSP.
Count Rejected Withdrawals Bank Card	Numeric	Number of rejected withdrawals not coming from a PSP.
Churn 7 Actual	Boolean	Label to be used during model training and testing.
First Deposit Date	Datetime	The first deposit date for the customer.

Table 4.3: Dataset variables and features from the industrial collaboration.

4.5.2 | Extraction Process

The industrial collaborator has a data warehousing system which follows the Kimball life cycle where first the source data is transferred to a staging area and then via their Extract Transform Load (ETL) process data is transferred to respective tables in the data warehouse (Alhajj et al., 2008). Data is summarised appropriately into different tables according to the business area (such as payments, business intelligence and so on).

Their data warehouse contains several fact tables which are used to store transactional data such as betting activity. Furthermore, it comprises of several dimension tables used for other types of data, for example, customer-related fields such as birth date, registration date and so on. As is seen in Table 4.3, there are a plethora of variables related to several main streams of data such as casino data and payment data. Such data was stored in different tables, and hence a separate Stored Procedure (SP) for each stream was used to aggregate the customer's five day data with respect to the FDD. The data from each SP was joined together based on the customer and stored in a denormalised format in a Comma Separated Value (CSV) file.

4.5.3 | Feature Engineering

The extracted raw data from the industrial partner data warehouse needed some amendments in order to be fitted to a machine learning model. The process mainly involved creating new features based on the raw features. For example, the *Verification Date* feature, which is the date and time on which the customer became a verified customer, that is completed the registration process was split up into other features. These features define the period of the day during which the verification process happened. This was done by using a one-hot encoding scheme where five different day period were generated mainly *Registration Night*, *Registration Morning*, *Registration Day 1*, *Registration Day 2* and *Registration Evening* features were created. The feature *Registration Night* would equate to one if the hour of verification occurred between one AM and seven AM otherwise zero. *Registration Morning* would equate to one if the hour of verification occurred between eight AM and noon otherwise zero. *Registration Day 1* is true if the verification date occurs between one PM and four PM otherwise false.*Registration Day 2* is true if the verification date occurs between five PM and seven PM otherwise false. Finally, *Registration Evening* would equate to one if the hour of verification occurred between eight PM and 11 PM otherwise zero. This effectively creates a sparse matrix where the feature was split into five different columns. The same was applied to the *Age* variable where different age bins were used such as *Age Bucket 24* if the customer's age is less than 24, *Age Bucket 25 28* if the customer's age is between 25 and 28, *Age Bucket 29 33* if the customer's age is between 29 and 33, *Age Bucket 34 40* if age is between 34 and 40, *Age Bucket 41 48* if age is between 41 and 48 and finally *Age Bucket 49* if the customer's age is greater than or equal to 49. Given these new features, the total number of features used during model training was 167. The most crucial feature which made this study possible is the First Deposit Date which was used to order the dataset accordingly.

4.5.4 | Machine Learning Models

As already briefly mentioned, the industrial collaborator currently employs an LGBM model to predict churn on their production environment. However, during selection, they also tested the RF model, which typically is the model of choice in churn prediction problems (Imran et al., 2019). Thus, for this work, the experiments shall be carried out using both RF and LGBM. Furthermore, an additional model was chosen termed XGBoost as according to Long et al. (2020) this model showed good performance in churn prediction.

4.5.5 | Technology Stack

In order to carry the experiments and data extraction previously explained, various tools were used. First, JetBrains DataGrip was used as the client to query the industrial collaborator data warehouse and SQL was used for querying. Secondly, for data analysis, the python library Pandas (version 1.0) was used. Thirdly, in order to implement the machine learning algorithms for the concept drift approaches, the python library of scikit-learn (version 0.23.1) was used. The experiments were executed on a single machine with a 4-core 3.6GHz CPU and 16GB of memory.

4.6 | Summary

This chapter highlighted the methodology and experiments which, firstly, can be used to highlight the presence of concept drift in the data, and secondly, how to tackle the problem of concept drift. The analysis started by using stationarity statistical tests such as ADF and KPSS. DT complexity experiments were also discussed that further highlight how concept drift can be monitored. Furthermore, the two main experiments of how to tackle concept drift, mainly the Moving Window Approach and Window Dissimilarity Approach were discussed.

5

Results & Evaluation

This chapter discusses the results that are obtained from the experiments performed on the industrial collaborator data. Firstly the experiments about analysing and identifying concept drift are exhibited. Secondly, the experiments from the two concept drift mitigation approaches, namely the Moving Window and Window Dissimilarity approaches, are presented. Thirdly the results are evaluated in view of the model performance obtained by the data science team from the industrial collaborator.

5.1 | Concept Drift Identification

Before addressing the problem of concept drift in churn prediction, several experiments were devised in order to identify concept drift. The investigation starts by listing the most important features, which are then used in the time series and decision tree complexity experiments. This was done to selectively choose from the plethora of features discussed in Section 4.5.1.

5.1.1 | Feature Importance

The feature importance experiment uses an Extra Trees (ET) classifier in order to work out the most important features in the dataset. The ET classifier uses several uncorrelated decision trees, and each tree selects the best feature to split the data. The features are selected by firstly ordering in a descending manner

according to the Gini Importance and secondly by selecting the top k features. Gini importance is a property of the tree which defines the importance for a particular feature. In this work, having the top 15 features was deemed adequate. The top 15 most significant features according to Gini importance are shown in a horizontal bar graph format in Figure 5.1. It can be noticed that *Number of Rounds Slots* and *Number of Rounds Mobile* are the top two features. The bar plot also shows that bet related features are more important than deposit related features.

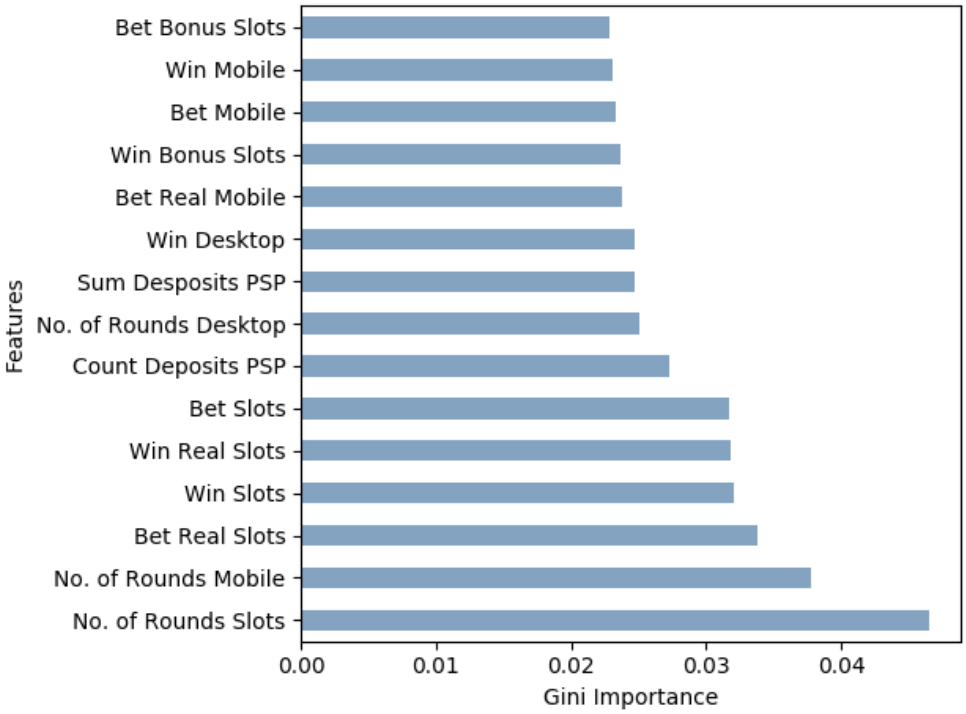


Figure 5.1: Extra trees feature importance bar plot show the top 15 features with *Number of Rounds Slots* being the most important feature with respect to churn prediction.

5.1.2 | Time Series Analysis

During the discussion of time series analysis, in section 4.3.1, two statistical methods, namely the Augmented Dickey Fuller (ADF) and the Kwiatkowski

Phillips Schmidt Shin (KPSS) are proposed to test for data stationarity. It discusses how a non-stationary time series indicates changing trends in the data. The null hypothesis of the ADF contrasts that of the KPSS as the null hypothesis of the ADF states there is a unit root (non-stationary) while that of KPSS states that the time series is trend stationary. Hence, if we fail to reject the null hypothesis of the ADF suggests non-stationarity, on the contrary, if we reject the null hypothesis of the KPSS suggests that the process has a unit root (non-stationary). Table 5.1 shows the p-value for both the ADF and KPSS for a number of variables (features). The selected features shown in Table 5.1, are not randomly selected from the whole feature set but are carefully chosen after analysing the results of the previous feature importance experiment.

Name	**ADF**	**KPSS**	**Outcome**
Number of Rounds Slots	**0.336**	**0.010**	Non-stationary
Number of Rounds Mobile	**0.329**	0.100	Non-stationary
Bet Real Slots	0.019	**0.010**	Non-stationary
Win Slots	**0.082**	**0.010**	Non-stationary
Win Real Slots	0.015	**0.010**	Non-stationary
Bet Slots	**0.092**	**0.010**	Non-stationary
Count Deposits PSP	**0.177**	0.100	Non-stationary
Number of Rounds Desktop	**0.106**	**0.010**	Non-stationary
Sum Deposits PSP	**0.065**	0.058	Non-stationary
Win Desktop	0.029	**0.010**	Non-stationary
Bet Real Mobile	0.000	**0.016**	Non-stationary
Win Bonus Slots	0.041	**0.010**	Non-stationary
Bet Mobile	0.000	**0.010**	Non-stationary
Win Mobile	0.000	**0.010**	Non-stationary
Bet Bonus Slots	0.041	**0.010**	Non-stationary

Table 5.1: P-values for ADF and KPSS Tests on selected features with bold values highlighting features that are non-stationary.

In the case of the ADF, variables that result in a p-value greater than the 0.05 alpha threshold imply a failure to reject the null hypothesis, hence implying non-stationarity. Contrastingly, in the case of KPSS, if the p-value is less than 0.05 alpha threshold, there is enough evidence for rejecting the null hypothesis in favour of the alternative, hence implying a unit root reflecting changing

trends (concept drift). From Table 5.1 it can be noted that some of the p-values are highlighted in bold. This was done to show which variables support the evidence for unit root and hence concept drift. It can also be noted that some of the features namely *Number of Rounds Slots*, *Win Slots*, *Bet Slots* and *Number of Rounds Desktop* portray stronger evidence of concept drift since both the ADF and KPSS suggested the presence of a unit root. Coincidentally, it was noted that the top feature *Number of Rounds Slots* exhibited the strongest p-value in the ADF test while also exhibiting the best feature importance metric. These results further corroborate the presence of concept drift and also shows how this feature is the most important in terms of predicting customer churn. For the full list of results for the ADF and KPSS stationarity tests one can refer to Table A.1 in Appendix A. Given the number of features present, inherently involves a plethora of statistical tests. Thus it can be argued that using the default p-value cutoff of 0.05, may result in having several tests that are statistically significant by chance. In order to cater for this effect, multiple correction testing methods such as the Bonferroni or Sidak methods may be used (Kononenko and Kukar, 2007). Using a multiple testing approach has the same effect as using the more stringent 0.01 alpha threshold (Zuo, 2019); however, in case of multiple testing, this is done on a stricter level as the p-value may be smaller than 0.01. Considering the Bonferroni method, the alpha threshold is divided by the number of statistical test present. This results in a p-value cutoff significantly lower than 0.05 (Kononenko and Kukar, 2007). In such a case, considering the ADF results, the majority of the features exhibit statistical significance further reinforcing the data variability or concept drift present. The same cannot be said for the KPSS results; however, having significant results from the ADF already shows the changing nature of the features.

5.1.3 | Decision Tree Complexity

The ADF and KPSS statistical tests indicate that the majority of the top 15 features are non-stationary and hence these tests suggest concept drift. To further reinforce this notion, a decision tree was used to gauge dataset complexity with the idea that if the decision tree is not limited in either of its parameters such

as depth and node count, one could effectively discern area of concepts that are represented by abnormal variation in the aforementioned parameters. The results achieved are shown in Figure 5.2.

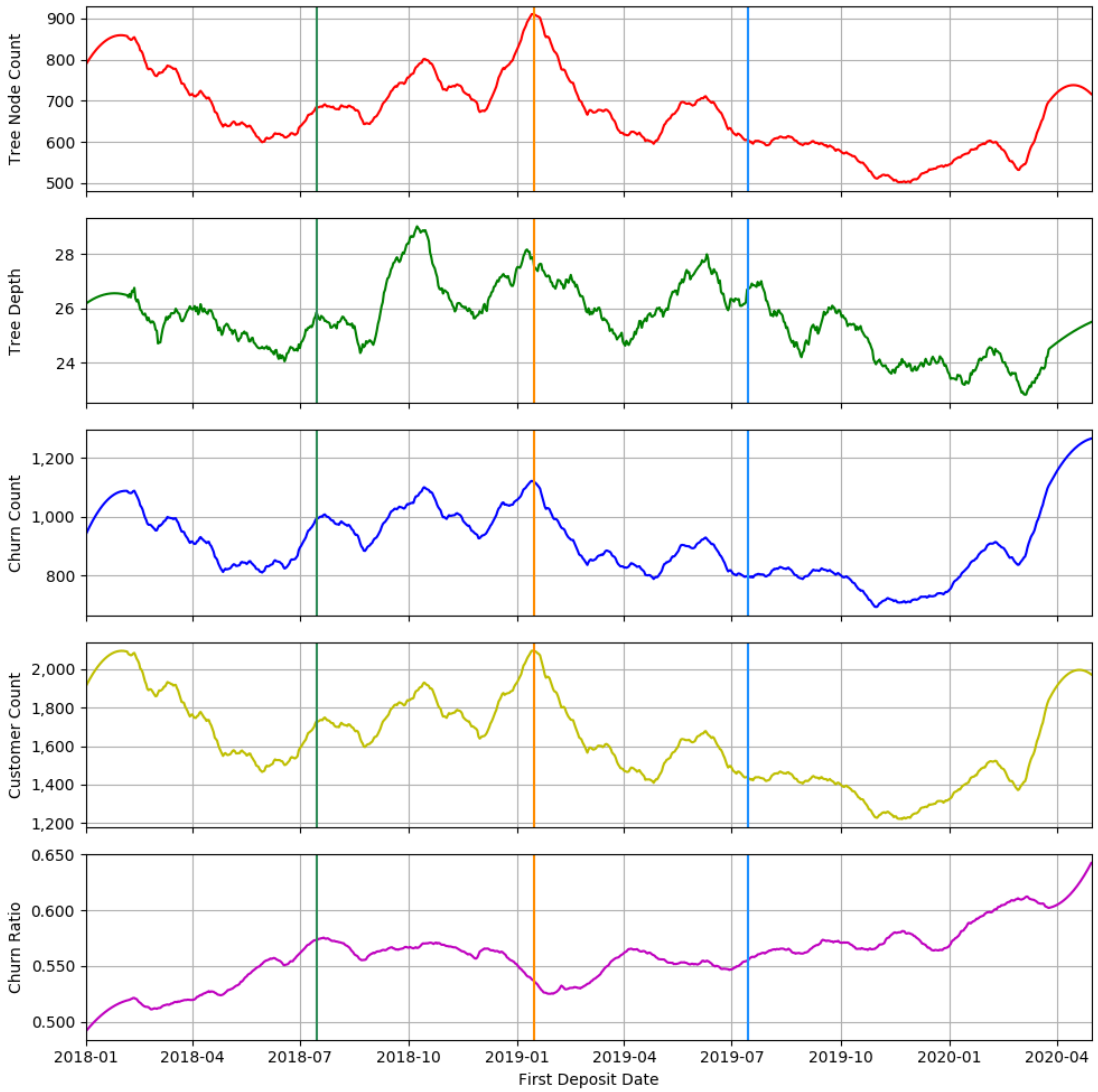


Figure 5.2: Decision tree node count and depth compared to customer count and churn properties for the first deposit date ordered dataset.

Five subplots were included being *tree node count*, *tree depth*, *churn count*, *customer count* and *churn ratio*. The *tree node count* and *tree depth* represent the number of nodes and depth respectively present in the tree during a particular

experiment. The *churn count* shows the number of churned customers during the experiment, while the *customer count* shows the total customers present in the experiment. As the name implies, the *churn ratio* is the ratio of the number of churned customers to the total customers present in a specific experiment. The inclusion of churn related data such as *churn count* was done to compare the tree's properties at instances where customer-related values showed sudden changes.

The churn ratio plot shows that for a particular first deposit date after 2018-07 (marked with a green vertical line in Figure 5.2) there was an increase in churn count. This increase in churn count happened after a continual decrease for the previous two month period (2018-05, 2018-06), which subsequently reflected in a rise of churn ratio. This change also displayed a slight rise in node count. Additionally, the churn ratio exhibited a decrease on a first deposit date in the region of 2019-01 (highlighted by an orange vertical line in Figure 5.2). Subsequently, this resulted in a sharp rise in node count, indicating that the dataset's underlying distribution sudden change.

The churn count also started decreasing at around 2019-07 (highlighted by a blue vertical line in Figure 5.2) which reflected in a decrease in node count and a slight increase in tree depth. However, in general, the tree depth remained constant, which points to the fact that the decision tree was not prone to overfitting. Overall, the plots follow a similar trend where a rise or drop in churn properties reflects a change in the tree's properties. The change reflected in the tree's properties does not necessarily follow the same pattern as the churn fields, that is if one increases the other one increases, further indicating evidence of changing trends in the underlying distribution of the data.

5.1.4 | Decision Tree Feature Importance

Given the decision tree complexity results, it may be deduced that if the decision tree's properties change according to these concepts, such changes may also be reflected in the feature importance. In order to manifest these changes in feature importance, the same experiment as the one shown in Section 5.1.3 was carried out, with a slight difference being that the feature importance (Gini value) for

several preselected features was monitored for each first deposit date in the ordered dataset. The outcome of this experiment is shown in Figure 5.3.

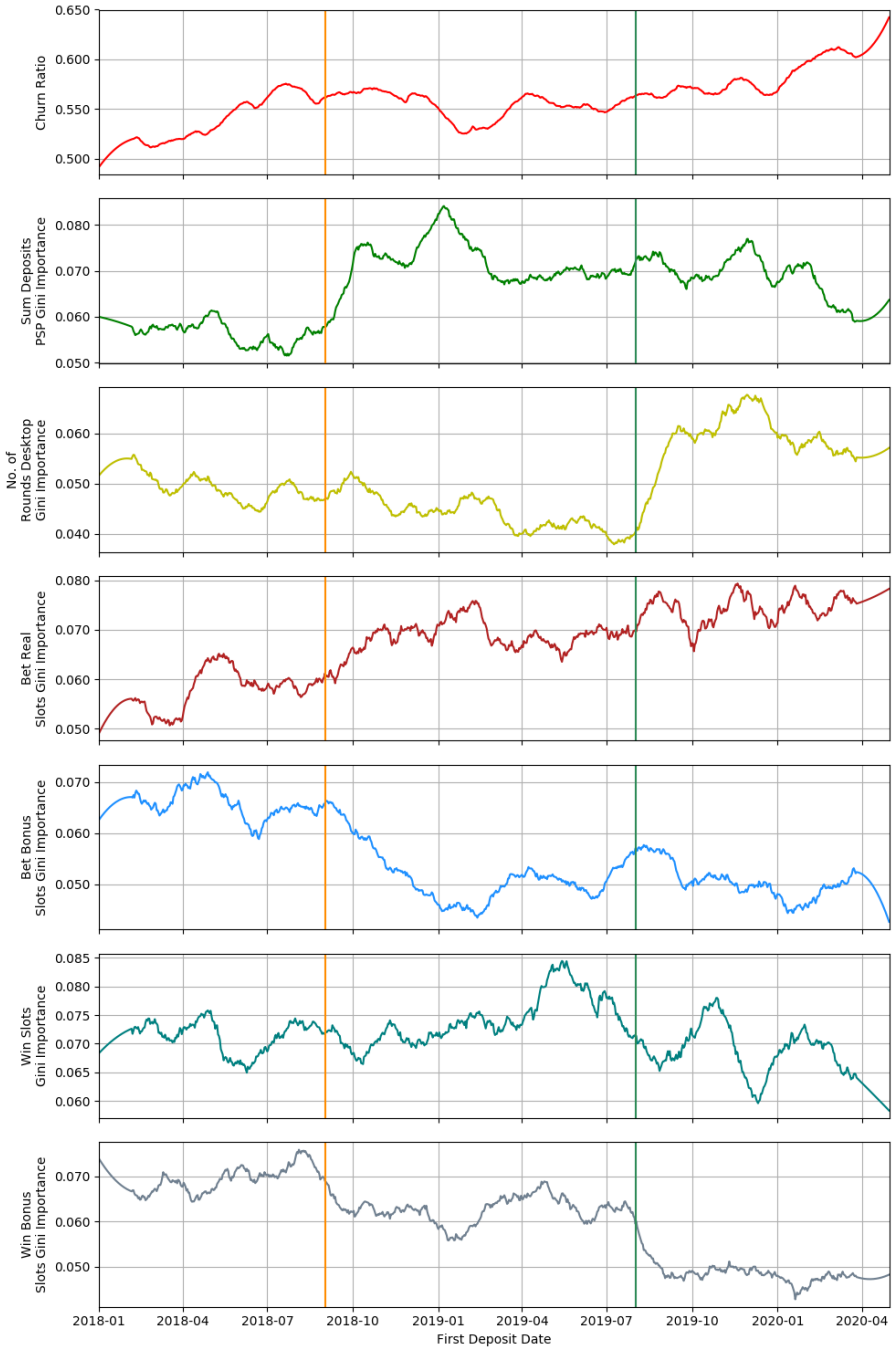


Figure 5.3: Decision tree feature importance variation for ordered dataset.

The selected features reflect some of the features from the feature importance plot shown in Figure 5.1. From Figure 5.3, it can be noted that in addition to feature importance plots, the churn ratio is also plotted to analyse the relationship between the various plots. It is evident that some of the features exhibit large shifts in their importance. In particular, the feature *No. of Round Desktop* which is the number of times a customer played from a desktop platform exhibits a change of around 50% at around the 2019-08 mark which is shown by a green vertical line. Furthermore, the importance of feature *Sum Deposits PSP* also shows a significant increase starting from 2018-09, which is highlighted by an orange line in Figure 5.3.

Consider for example *Win Bonus Slots* with respect to *No. of Round Desktop* between the 2019-07 and 2019-10 period where the former shows a significant decrease while the latter exhibits a sharp rise. It can be observed that the *Win Bonus Slots* importance decreases by around 30% while the *No. of Round Desktop* portrays a boost in importance. Furthermore, this growth in importance (*No. of Round Desktop*) happens while the churn ratio shows an upward trend. Through these plots, it can be noted that the effects of these concepts and changing trends are translated into variations in feature importance.

5.2 | Learning with Concept Drift

According to Dong et al. (2019) and Gama et al. (2009b), the sequential predictive approach is the preferred option for evaluating datasets suffering from data drifts. For this work, this approach was adopted where each window will result in a particular value for the selected metric. In order to achieve the final value, the average for all the windows is computed. The data science team presented us with ROC-AUC as the performance metric to be used for comparison. However, in order to monitor the performance of the machine learning model, other metrics were included in this work to have a better understanding of the model's performance. In particular, the confusion matrix was also noted as that can be used to extract precision and recall metrics (amongst others). The research of Abou Trab et al. (2015), shows that customer churn prediction is primarily in-

terested in reducing the incorrect classification of churners as this leads to profit loss. Hence, through the use of individual confusion matrices, one can compute metrics such a recall (sensitivity). Since, in churn prediction, models with high sensitivity are preferred than those with high specificity (Abou Trab et al., 2015).

The dataset used in the following experiments contained two and a half years worth of data which resulted in approximately 1.4 million customers. For the different methods listed in Section 4.4, several parameters were varied, namely the training window size, the testing window size and moving window step. In this case, the value in each of these parameters refers to a predefined number of rows in the ordered data; these windows are also commonly referred to as block size in literature (Castillo et al., 2004). For all the experiments, the dataset was split into 75% and 25% with the 25% portion being used for evaluation. The test window sizes used were that of 1,000 and 5,000 rows, and the training window sizes were chosen to be 50,000, 100,000 and 150,000 rows. Note that a row refers to a unique customer, and hence can be used interchangeably. Having 50,000 rows amounts to around 15 days worth of data depending on the customer activity. For all the tests, the window is shifted by 1,000 rows each iteration, which according to the customer count plot in Figure 5.2 equates to around a day's worth of data.

5.2.1 | Moving Window Approach

In the moving window approach, a new model is retrained with every window independent of the fact that there might not be any difference in data distribution and or data concepts. The moving window model performs a prediction on a future adjacent testing window. During each window, a model which is trained once on a specific portion of the dataset is also used to perform prediction on the same testing window; this is known as the static model. Thus for each testing window encountered two models are used for prediction, the moving window and the static model. This was done so that one can evaluate performance improvement. Furthermore, as briefly mentioned, different training portions were used for the static model, being at 20%, 50% and 75% with the last option replicating the work of the industrial partner data science team. For

all the experiments, the model evaluation was performed on the last 25% of the first deposit date ordered dataset.

Figure 5.4 depicts the resulting ROC-AUC plot when using a Random Forest (RF) model, a training window of 150,000 rows (or customers) and testing window of 5,000 rows (or customers) while training the static model on the first 20% portion of the data. From the ROC-AUC plot shown it can be noted that for every testing window the moving window approach model (blue) outperformed the static model (orange). In contrast, both models show a monotonically increasing trend. This already indicates how the moving window technique can be used to improve prediction performance under circumstances where the dataset suffers from severe data variability. Interestingly, 20% of the dataset equates to almost 300,000 rows or customers, which is effectively twice the data used in the moving window model. Thus, albeit trained on more data, the static model did not manage to surpass the moving window in any of the testing windows shown.

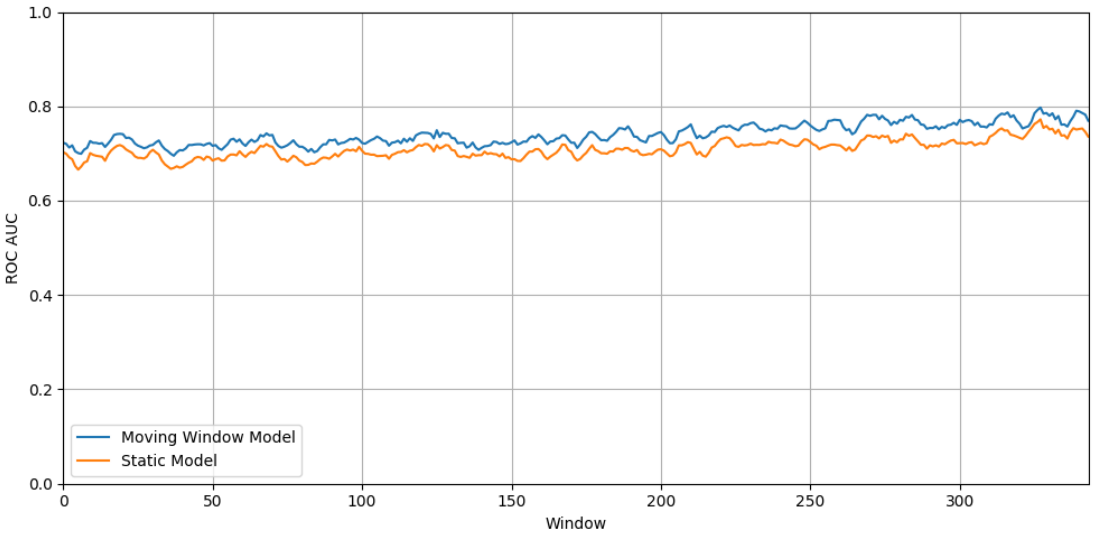


Figure 5.4: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on first 20% of the ordered dataset.

Figure 5.5 illustrates the recall or sensitivity plot when using a Random Forest (RF) model, a training window of 150,000 rows (or customers) and testing window of 5,000 rows (or customers) while training the static model on the first 20% portion of the data. When examining the recall plot (Figure 5.5), the same behaviour as shown in the previous ROC-AUC plot is noted where the moving window approach model outperforms the static model.

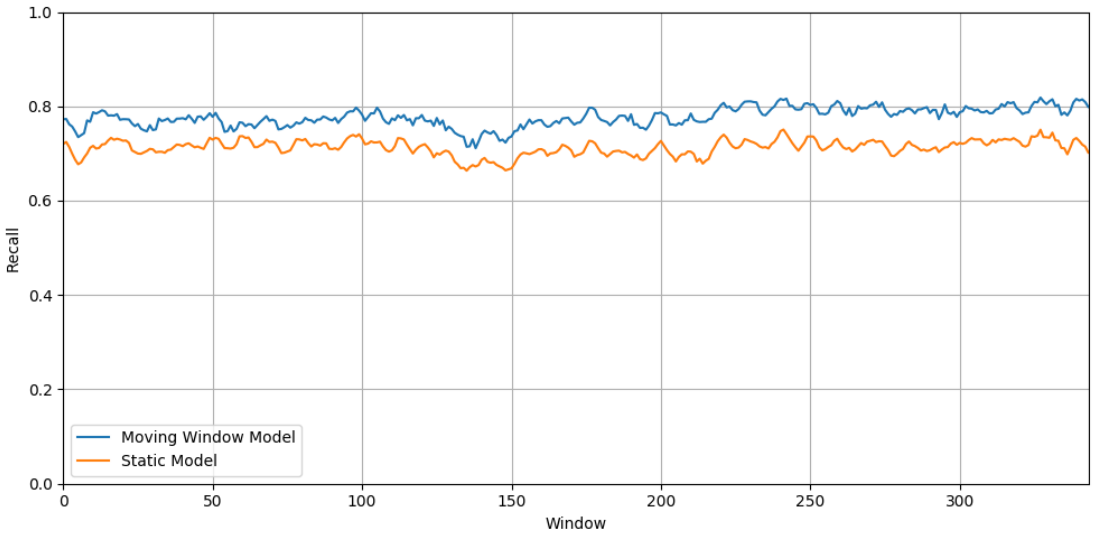


Figure 5.5: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on first 20% of the ordered dataset.

Figure 5.6 and Figure 5.7 display the ROC-AUC and recall plot respectively, when using Random Forest (RF) model, a training window of 150,000 and testing window of 5,000 and the static model being trained on the first 50% of the first deposit date ordered dataset. In this scenario, the moving window model still outperforms the static model. However, one can note that there is a small difference between the ROC-AUC plots of the two models up until the first quarter (till around window 125) of the testing windows. This is due to the static model being trained on around 700,000 customers and hence manages to learn more from the overall data available. The recall plot in Figure 5.7 exhibits this behaviour stronger, as up until the first quarter of the testing windows the two models are almost indistinguishable with moving window model having a slight edge in the majority of the first quarter. This behaviour indicates that

from around testing window 125 onwards the moving window is learning new concepts which are not part of the training data of the static model. Hence, inevitably, the moving window model outperforms the static model.

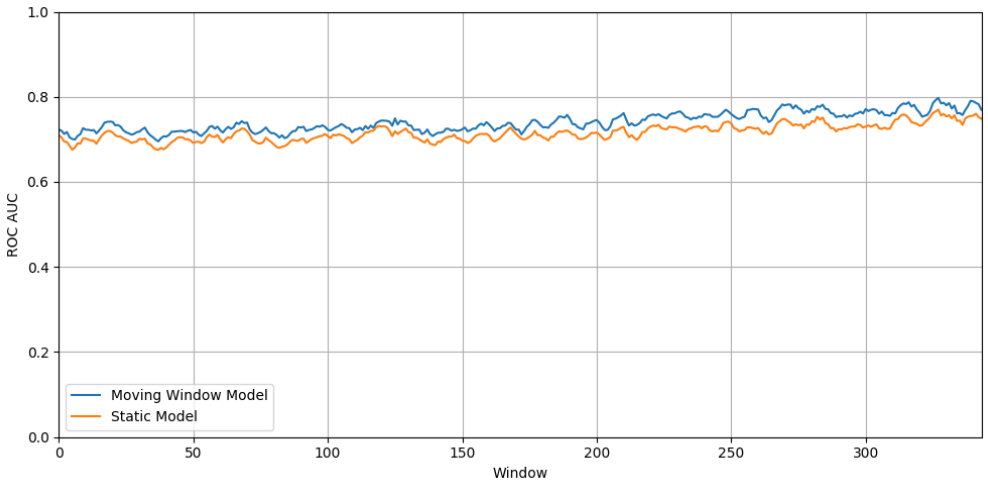


Figure 5.6: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on first 50% of the ordered dataset.

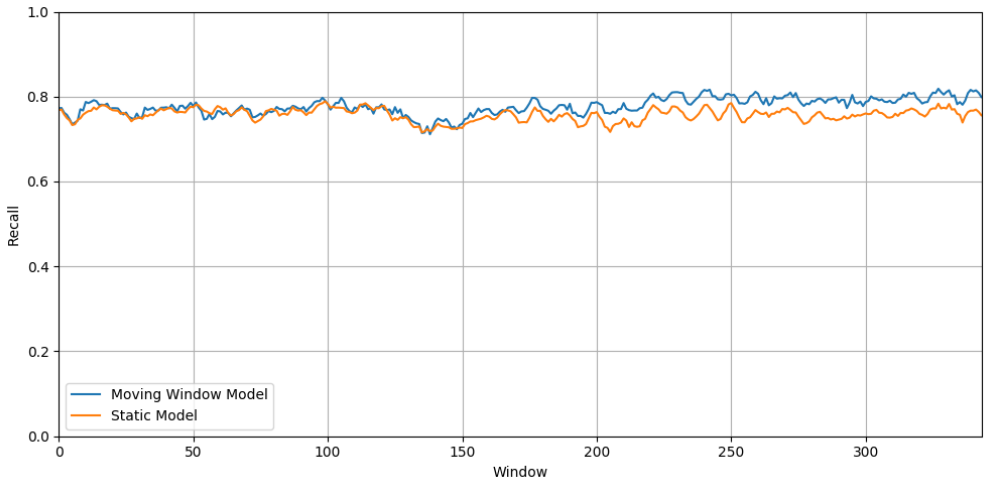


Figure 5.7: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on first 50% of the ordered dataset.

Figure 5.8 and Figure 5.9 show plots of ROC-AUC and recall respectively, when using a Random Forest (RF) model, training window size of 150,000 and

testing window of 5,000 while using the first 75% of the first deposit date ordered dataset to train the static model.

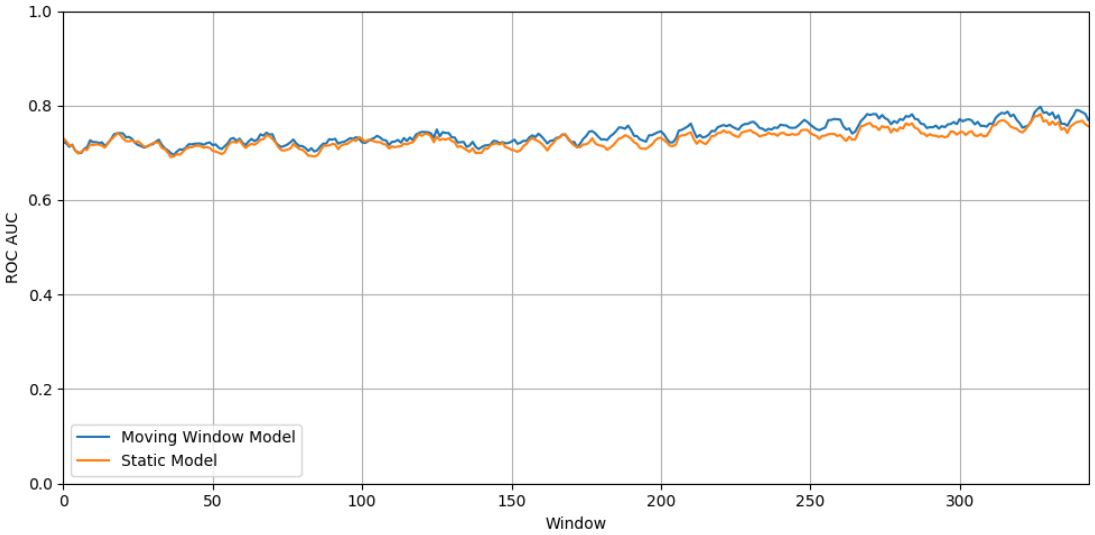


Figure 5.8: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on first 75% of the dataset.

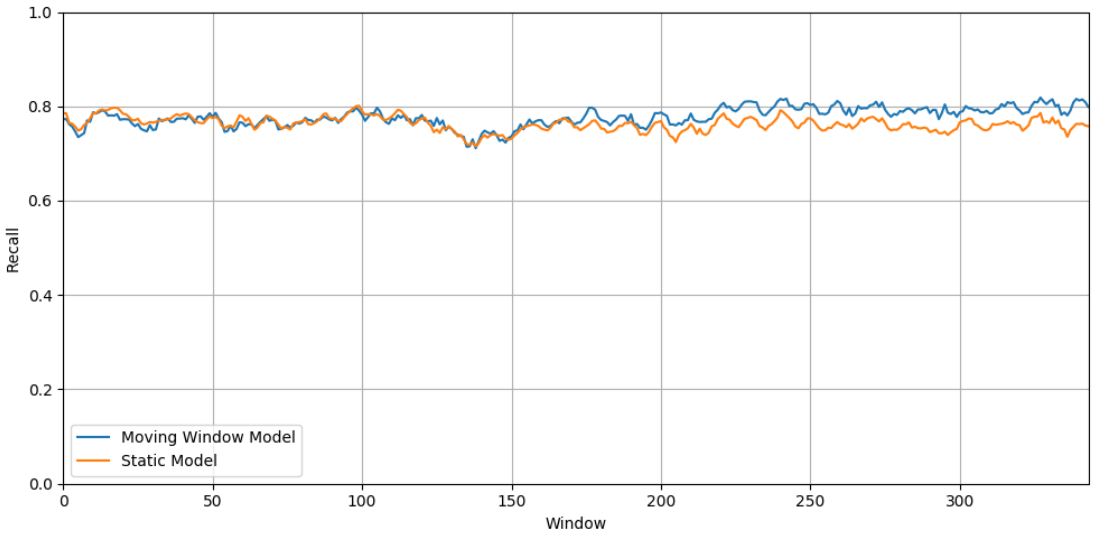


Figure 5.9: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on first 75% of the dataset.

Analysing the ROC-AUC plot (Figure 5.8), it can noted that at the beginning the two models are almost identical with the moving window model having a slight edge. This indicates that the static model can learn the majority of the concepts during the first quarter of the evaluation portion. However, as new concepts are introduced in the dataset, the moving window approach model can assimilate all these concepts. Thus able to pull away in terms of predictive performance when testing on windows that are far away from the period on which the static model is trained. The same behaviour can be noted for the recall plot in Figure 5.9.

5.2.2 | Window Dissimilarity Approach

The window dissimilarity approach performs model updates intelligently and uses a proactive rather than a passive approach, in the sense that on each window, the model is not blindly retrained. However, a statistical test is used to determine if there were any data distribution changes. Suppose the result shows that the difference in data distribution is statistically significant. In that case, the previous model is discarded, and a new model is trained on the data of the current window only. In order to be able to evaluate with the previous approach the same testing structure was adopted, that is the window dissimilarity model evaluated against the static model (trained on 20%, 50% and 75% of the dataset). The window dissimilarity model and the static model are evaluated on the last 25% of the dataset.

Figure 5.10 depicts the resulting ROC-AUC when using a Random Forest (RF) model, a training window of 150,000 and testing window of 5,000 while training the static model on the first 20% portion of the first deposit date ordered dataset. It can be noted that the window dissimilarity model outperforms the static model in every testing window available. An important point to keep in mind is that in the window dissimilarity approach, the model is not necessarily retrained on every window, as the dissimilarity measure controls this. In this case, the dissimilarity metric results from the Kolmogorov-Smirnov (KS) statistical test. However, at worst, the moving window approach and the window dissimilarity approach become equivalent if the changing trends (concepts)

occur in every window. If this occurs the model needs to be retrained in the window dissimilarity approach; since during each window the dissimilarity metric results to be statistically significant, signalling a retraining event.

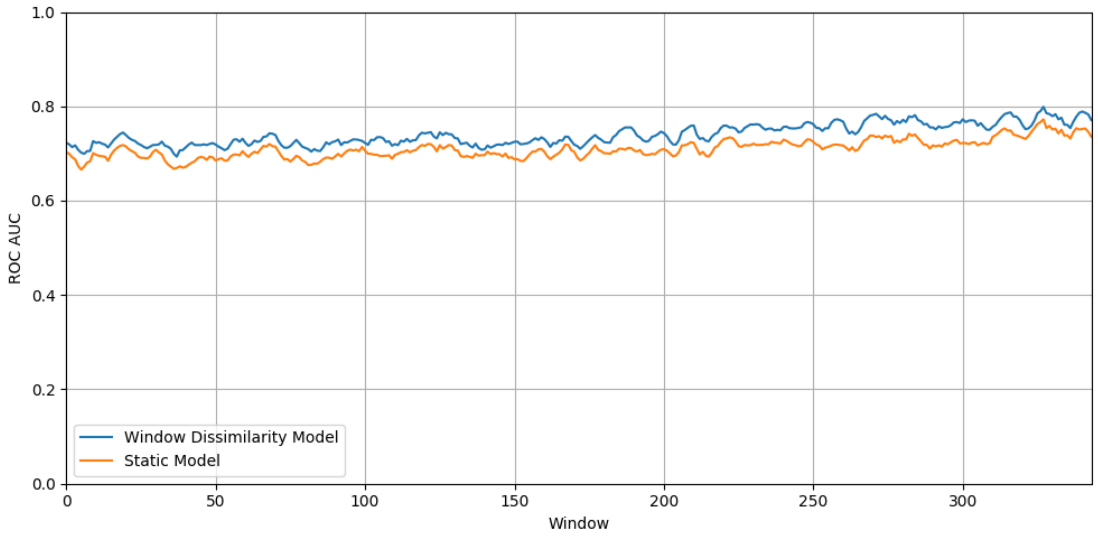


Figure 5.10: Window Dissimilarity Approach: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on first 20% of the dataset.

Figure 5.11 shows the recall or sensitivity plot when using a Random Forest (RF) model, a training window of 150,000 and testing window of 5,000 while training the static model on the first 20% portion of the first deposit date ordered dataset. From Figure 5.11, it can be noted that the window dissimilarity approach model shows instances where the recall metrics results to be slightly above 0.8. In particular, this happens from around window 220 onwards. The same cannot be said for the static model as it does not show a monotonically increasing trend. This further reinforces the notion that when using a static model, the model's staying power tends to diminish as the model predicts distant periods from the training period. Similar to the result obtained from the moving window approach (Figure 5.5), it can be noted that for every testing window, the window dissimilarity approach model (blue) outperforms the static model (orange).

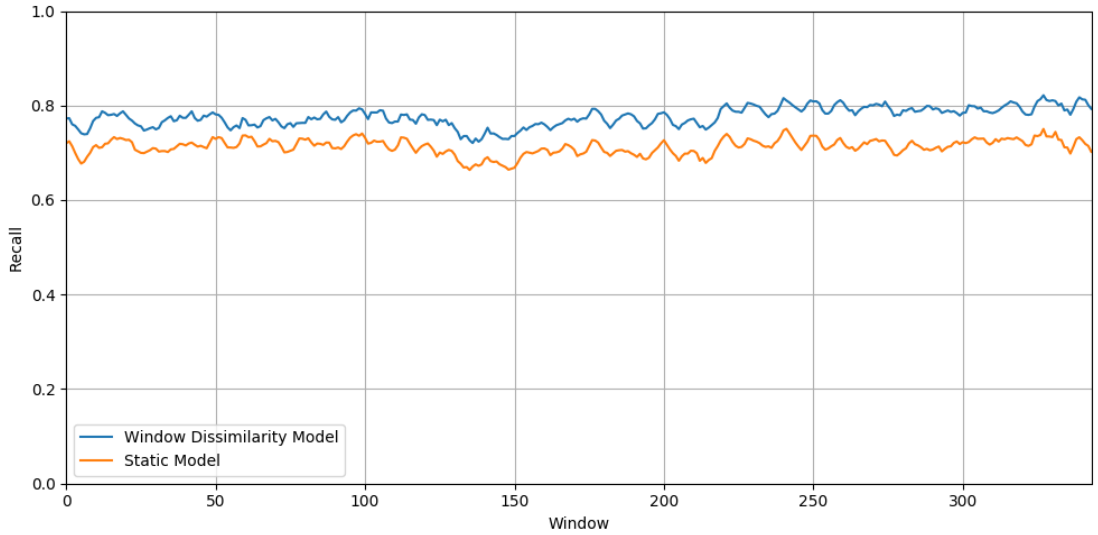


Figure 5.11: Window Dissimilarity Approach: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on the first 20% of the dataset.

Figure 5.12 and Figure 5.13 illustrate the ROC-AUC and recall plots as discussed before that is when using Random Forest (RF) model, however for a training window of 150,000 and testing window of 5,000 and the static model being trained on the first 50% of the data. In this scenario, the window dissimilarity approach model still outperforms the static model; however, it can be noted that the gap between the two is smaller at the beginning of the evaluation period. Furthermore, in the case of the recall plot (Figure 5.13) for the first quarter of the evaluation period, the two models exhibit similar performance. This behaviour is as seen in the moving window approach; attributed to the fact that the static model can learn more from the changing trends in the dataset and thus be able to close the performance gap between the window dissimilarity approach model. However, despite the static model having a significantly larger exposure to data, still is surpassed by the window dissimilarity approach.

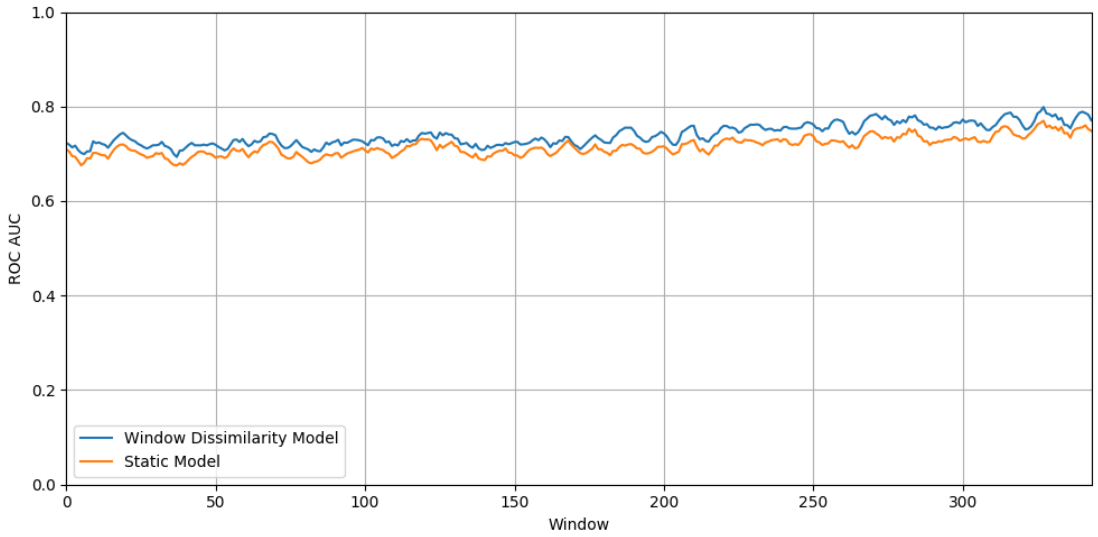


Figure 5.12: Window Dissimilarity Approach: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on first 50% of the dataset.

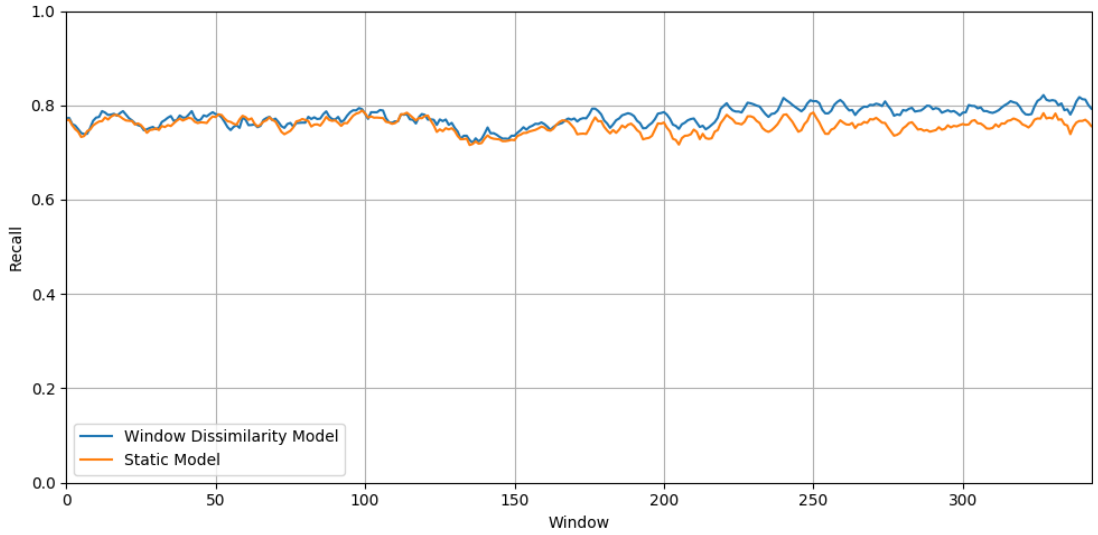


Figure 5.13: Window Dissimilarity Approach: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on first 50% of the dataset.

Figure 5.14 and Figure 5.15 show the plots for ROC-AUC and recall respectively, for the window dissimilarity approach when using a Random Forest (RF) model, training window size of 150,000 and testing window of 5,000 while the static model is trained on the first 75% of the first deposit date ordered dataset. In this case, the window dissimilarity approach exhibited similar performance to the static model during the first portion of the evaluation period as both attain similar ROC-AUC scores.

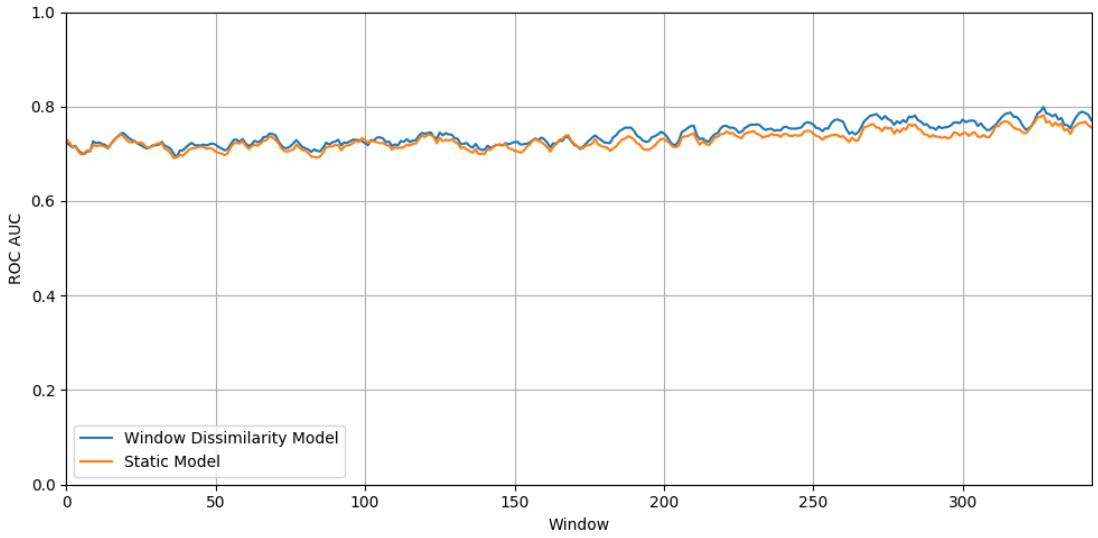


Figure 5.14: Window Dissimilarity Approach: RF ROC-AUC plot using training window of 150,000, testing window of 5,000 and static model trained on the first 75% of the dataset.

However, as new concepts are encountered during the second half of the evaluation period, the window dissimilarity approach outperforms the static model. The same behaviour is reflected in the recall plot (Figure 5.15). Despite being trained on the first 75% of the first deposit date ordered dataset, the static model did not manage to outperform the window dissimilarity approach. The first 75% accounts to around one million customers (rows) which results to be seven times as much as the data used by the window dissimilarity approach model. Clearly, the window dissimilarity approach offers better performance while drastically reducing the training times required.

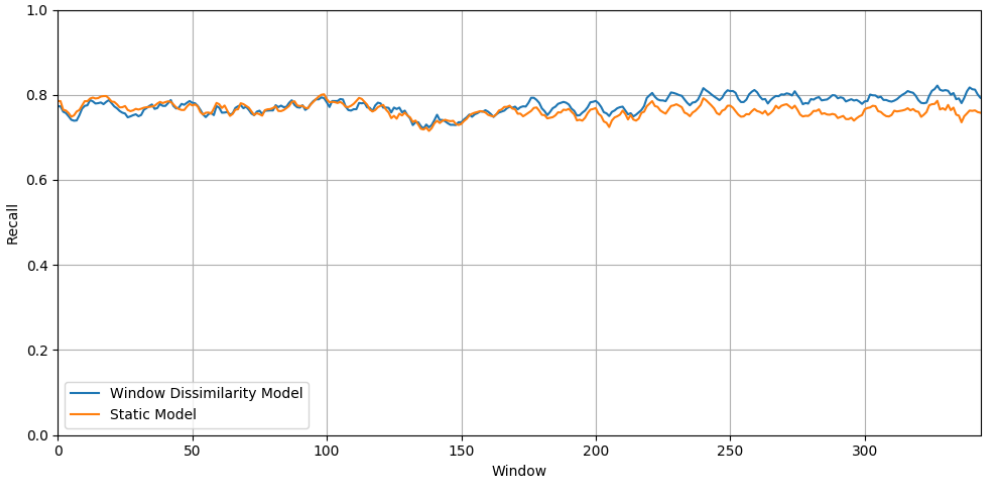


Figure 5.15: Window Dissimilarity Approach: RF recall plot using training window of 150,000, testing window of 5,000 and static model trained on first 75% of the dataset.

5.3 | Evaluation

As seen in the last two sections, the results for the two concept drift approaches pertaining to the RF model were discussed. However, the plots only show a specific parameter combination; that of training window of 150,000 rows (customers) and testing window of 5,000 rows (customers). For evaluation purposes, all the parameter combination for all the machine learning models are tabulated. In this study, three machine learning models were used. The first being RF whose results are tabulated in Table 5.2 for the moving window approach and Table 5.5 for the window dissimilarity approach. The results pertaining to LGBM are tabulated in Table 5.3 for the moving window approach and the window dissimilarity approach are shown in Table 5.6. Lastly for the XGBoost model, the results are tabulated in Table 5.4 for the moving window approach and in Table 5.7 for the window dissimilarity approach. Each of the tables mentioned contains all the parameter combinations used for evaluation purposes, namely the three static model training sizes, the three concept drift approaches training sizes and two testing window sizes. Bold values in tables accentuate highest scores.

Static	Moving Window		Static	Moving Window
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1,000	0.709	0.736
20%	50,000	5,000	0.709	0.736
20%	100,000	1,000	0.709	0.739
20%	100,000	5,000	0.710	0.739
20%	150,000	1,000	0.709	0.740
20%	150,000	5,000	0.710	**0.741**
50%	50,000	1,000	0.714	0.740
50%	50,000	5,000	0.715	0.740
50%	100,000	1,000	0.714	0.740
50%	100,000	5,000	0.715	0.740
50%	150,000	1,000	0.714	**0.741**
50%	150,000	5,000	0.715	**0.741**
75%	50,000	1,000	0.725	0.736
75%	50,000	5,000	0.726	0.737
75%	100,000	1,000	0.725	0.739
75%	100,000	5,000	0.726	0.739
75%	150,000	1,000	0.725	0.740
75%	150,000	5,000	0.726	0.740

Table 5.2: Moving Window: Average ROC-AUC score for RF.

Static	Moving Window		Static	Moving Window
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1,000	0.727	0.752
20%	50,000	5,000	0.727	0.751
20%	100,000	1,000	0.727	0.754
20%	100,000	5,000	0.727	0.754
20%	150,000	1,000	0.727	0.755
20%	150,000	5,000	0.727	**0.755**
50%	50,000	1,000	0.731	0.752
50%	50,000	5,000	0.731	0.751
50%	100,000	1,000	0.731	0.754
50%	100,000	5,000	0.731	0.754
50%	150,000	1,000	0.731	**0.755**
50%	150,000	5,000	0.731	**0.755**
75%	50,000	1,000	0.740	0.752
75%	50,000	5,000	0.740	0.752
75%	100,000	1,000	0.740	0.754
75%	100,000	5,000	0.740	0.754
75%	150,000	1,000	0.740	0.755
75%	150,000	5,000	0.740	0.755

Table 5.3: Moving Window: Average ROC-AUC score for LGBM.

Static	Moving Window		Static	Moving Window
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1,000	0.723	0.745
20%	50,000	5,000	0.723	0.744
20%	100,000	1,000	0.723	0.751
20%	100,000	5,000	0.723	0.750
20%	150,000	1,000	0.723	0.752
20%	150,000	5,000	0.723	**0.752**
50%	50,000	1,000	0.730	0.745
50%	50,000	5,000	0.730	0.744
50%	100,000	1,000	0.730	0.751
50%	100,000	5,000	0.730	0.750
50%	150,000	1,000	0.730	**0.752**
50%	150,000	5,000	0.730	**0.752**
75%	50,000	1,000	0.741	0.745
75%	50,000	5,000	0.741	0.745
75%	100,000	1,000	0.742	0.751
75%	100,000	5,000	0.742	0.750
75%	150,000	1,000	0.742	0.752
75%	150,000	5,000	0.742	0.752

Table 5.4: Moving Window: Average ROC-AUC score for XGBoost.

Static	Window Dissimilarity		Static	Window Dissimilarity
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1000	0.709	0.735
20%	50,000	5000	0.709	0.735
20%	100,000	1000	0.709	0.738
20%	100,000	5000	0.710	0.738
20%	150,000	1000	0.709	0.739
20%	150,000	5000	0.710	**0.740**
50%	50,000	1000	0.714	0.735
50%	50,000	5000	0.715	0.735
50%	100,000	1000	0.714	0.738
50%	100,000	5000	0.715	0.738
50%	150,000	1000	0.714	0.739
50%	150,000	5000	0.715	**0.740**
75%	50,000	1000	0.725	0.735
75%	50,000	5000	0.726	0.737
75%	100,000	1000	0.725	0.738
75%	100,000	5000	0.726	0.738
75%	150,000	1000	0.725	0.739
75%	150,000	5000	0.726	0.740

Table 5.5: Window Dissimilarity: Average ROC-AUC score for RF.

Static	Window Dissimilarity		Static	Window Dissimilarity
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1000	0.727	0.751
20%	50,000	5000	0.727	0.751
20%	100,000	1000	0.727	0.753
20%	100,000	5000	0.727	0.753
20%	150,000	1000	0.727	0.754
20%	150,000	5000	0.727	**0.754**
50%	50,000	1000	0.731	0.751
50%	50,000	5000	0.731	0.751
50%	100,000	1000	0.731	0.753
50%	100,000	5000	0.731	0.753
50%	150,000	1000	0.731	**0.754**
50%	150,000	5000	0.731	**0.754**
75%	50,000	1000	0.740	0.751
75%	50,000	5000	0.740	0.751
75%	100,000	1000	0.740	0.753
75%	100,000	5000	0.740	0.753
75%	150,000	1000	0.740	0.754
75%	150,000	5000	0.740	0.754

Table 5.6: Window Dissimilarity: Average ROC-AUC score for LGBM.

Static	Window Dissimilarity		Static	Window Dissimilarity
Training Size	Training Size	Testing Size	AUC	AUC
20%	50,000	1000	0.723	0.744
20%	50,000	5000	0.723	0.743
20%	100,000	1000	0.723	0.750
20%	100,000	5000	0.723	0.750
20%	150,000	1000	0.723	0.752
20%	150,000	5000	0.723	**0.752**
50%	50,000	1000	0.730	0.744
50%	50,000	5000	0.730	0.743
50%	100,000	1000	0.730	0.750
50%	100,000	5000	0.730	0.750
50%	150,000	1000	0.730	**0.752**
50%	150,000	5000	0.730	**0.752**
75%	50,000	1000	0.741	0.744
75%	50,000	5000	0.741	0.743
75%	100,000	1000	0.742	0.750
75%	100,000	5000	0.742	0.750
75%	150,000	1000	0.742	0.752
75%	150,000	5000	0.742	0.752

Table 5.7: Window Dissimilarity: Average ROC-AUC score for XGBoost.

5.3.1 | Industrial Collaborator

The data science team of the industrial collaborator has a running production model for churn prediction. Their machine learning pipeline consists of first using cross-validation techniques for model selection. Then in order to productionise the churn prediction model a train-test split procedure of 75% training and 25% testing; prior to the dataset split, the dataset is shuffled. The data science team selected two models being the RF and LGBM with the performance metric of choice being ROC-AUC, hence the reason why this study uses the ROC-AUC as the primary performance metric amongst others such as recall. Given the previous tables shown, the best results for all the different machine learning models and both approaches are tabulated in Table 5.8. Additionally, Table 5.8 also contains the performance metric of the industrial collaborator. It can be noted that the results for XGBoost are only present for the two approaches used in this study as the industrial collaborator did not use this model. However, XGBoost is evaluated in this study, due to the popularity and performance in churn prediction research (Do et al., 2017). Furthermore, when comparing the results achieved by the two concept drift approaches with those achieved by the industrial collaborator, it can be noted that there is approximately 3% performance increase.

Approach	RF	LGBM	XGBoost
Moving Window	0.741	0.755	0.752
Window Dissimilarity	0.740	0.754	0.752
Industrial Collaborator	0.721	0.739	N/A

Table 5.8: Evaluation results summary using the ROC-AUC metric.

5.3.2 | Concept Drift Approaches

Apart from the evaluation with the results achieved by the industrial collaborator's data science team, a comparison between the static model and the moving window model approaches is made. This can also be carried out by looking at the tabulated results in Table 5.2 and Table 5.5 where the results from RF are showcased. The performance improvement for the RF model, in particular,

ranged from 2.00 to 4.22% depending on the amount of data the static model was trained on. This clearly indicates that if data suffers from changing trends or concepts, employing a moving window technique is beneficial. The other machine learning achieved similar performance improvement.

Further to such comparisons, it is better to evaluate the statistical significance of the results achieved to verify that such approaches portray statistically sound results, albeit showing a clear percentage improvement. In this case, such statistical significance tests are possible because the last 25% portion of the data consists of multiple testing windows. Each testing window contains two results reflecting the performance of the concept drift approach and the static approach, respectively.

This resulting format makes it a perfect fit for a paired sample study, where the before and after results of applying concept drift approaches can be scrutinised. In addition to defining the hypothesis, a decision on the most appropriate statistical test needs to be taken. The results achieved are of the quantitative type. Thus the paired sample statistical tests can be narrowed down to either the Wilcoxon Signed Rank test (non-parametric) or the Paired Sample *t* test (parametric). The choice between the two tests depends on whether the sample follows a normal distribution or not. Due to the sample size, the Shapiro Wilk Normality test was chosen and for the different test scenarios, if the resulting p-value is less than 0.05 the null hypothesis that the sample is normally distributed can be rejected. Table 5.9 shows the results from the normality test. It can be noted that the majority of the result samples are not normally distributed as the resulting p-value from the Shapiro Wilk normality test is less than 0.05.

Consequently, the Wilcoxon Signed Rank test is used to test whether the implementation of the concept drift approaches significantly affects the predictive performance. The results for the Wilcoxon Signed Rank Test are tabulated in Table 5.10. It can be noted that all the p-values are less than the 0.05 significance level. It is important to note that all the p-values for the Wilcoxon Signed Rank test resulted in being of negative exponent (for example 3.880e-58) thus all round to zero. Hence, there is a statistically significant difference between the results achieved by the static model and those achieved with the windowing approach. This statistical test further reinforces the effectiveness of the windowing

Static	Moving Window		Static	Moving Window	Normality
Training Size	Training Size	Testing Size	P-value	P-value	Outcome
20%	50,000	1,000	0.329	**0.029**	Not Normal
20%	50,000	5,000	**0.001**	**0.000**	Not Normal
20%	100,000	1,000	0.329	**0.028**	Not Normal
20%	100,000	5,000	**0.001**	**0.000**	Not Normal
20%	150,000	1,000	0.329	**0.036**	Not Normal
20%	150,000	5,000	**0.001**	**0.000**	Not Normal
50%	50,000	1,000	0.716	**0.029**	Not Normal
50%	50,000	5,000	**0.000**	**0.000**	Not Normal
50%	100,000	1,000	0.716	**0.028**	Not Normal
50%	100,000	5,000	**0.000**	**0.000**	Not Normal
50%	150,000	1,000	0.716	**0.036**	Not Normal
50%	150,000	5,000	**0.000**	**0.000**	Not Normal
75%	50,000	1,000	0.146	**0.029**	Not Normal
75%	50,000	5,000	**0.001**	**0.000**	Not Normal
75%	100,000	1,000	0.146	**0.028**	Not Normal
75%	100,000	5,000	**0.001**	**0.000**	Not Normal
75%	150,000	1,000	0.146	**0.036**	Not Normal
75%	150,000	5,000	**0.001**	**0.000**	Not Normal

Table 5.9: Shapiro Wilk Normality tests for RF ROC-AUC score samples.

Static	Moving Window		Significance	Wilcoxon
Training Size	Training Size	Testing Size	P-value	Outcome
20%	50,000	1,000	**0.000**	Significant Improvement
20%	50,000	5,000	**0.000**	Significant Improvement
20%	100,000	1,000	**0.000**	Significant Improvement
20%	100,000	5,000	**0.000**	Significant Improvement
20%	150,000	1,000	**0.000**	Significant Improvement
20%	150,000	5,000	**0.000**	Significant Improvement
50%	50,000	1,000	**0.000**	Significant Improvement
50%	50,000	5,000	**0.000**	Significant Improvement
50%	100,000	1,000	**0.000**	Significant Improvement
50%	100,000	5,000	**0.000**	Significant Improvement
50%	150,000	1,000	**0.000**	Significant Improvement
50%	150,000	5,000	**0.000**	Significant Improvement
75%	50,000	1,000	**0.000**	Significant Improvement
75%	50,000	5,000	**0.000**	Significant Improvement
75%	100,000	1,000	**0.000**	Significant Improvement
75%	100,000	5,000	**0.000**	Significant Improvement
75%	150,000	1,000	**0.000**	Significant Improvement
75%	150,000	5,000	**0.000**	Significant Improvement

Table 5.10: Wilcoxon Signed Rank tests for RF ROC-AUC score samples.

techniques proposed in Section 4.4. Having, such performance improvement is of great benefit for the industrial collaborator due to the potential improvement in customer retention, ultimately reeling in more profit.

5.4 | Summary

In this chapter, the results of the two concept drift approaches were presented. Firstly the result of the feature importance experiment was presented, which shows that the top two features are *Number of Rounds Slots* and *Number of Rounds Mobile*. Secondly, the results from two statistical test, namely ADF and KPSS show how several features portray non-stationarity; indicating changing trends or concepts. Thirdly, results of the decision tree complexity were presented which show, how the tree node count and depth varies to other variables such as *churn count*. The results corroborate, those from the statistical analysis, further reinforcing the notion of changing trends or concepts. Lastly, the results for the two approaches being the moving window and window dissimilarity approach are discussed. The concept drift approaches were tested using three different machine learning models being RF, LGBM and XGBoost. Results show that the concept drift approaches attain a performance improvement of 2.00 to 4.22% when considering RF and the various static models used. When evaluating these approaches with the results achieved by the industrial collaborator data science team, an improvement of around 3% is noted. The chapter concludes by gauging whether the results achieved are statistically significant or not by using a paired sample statistical test; all results show a statistically significant improvement.

6

Conclusion

This chapter synthesises, the work carried out on churn prediction with particular focus to concept drift. This work comprised of two parts, the first being reviewing and investigating the work of Choi et al. (2017) and the second involved the industrial collaboration with a local iGaming company.

6.1 | Achieved Aims and Objectives

The first part of this work started by investigating churn prediction in online mobile games by reviewing the research done by Choi et al. (2017). Prior to beginning any analysis, an appropriate dataset had to be extracted from the raw dataset. This involved creating a number of features such as *mean_score* and *active_duration*. Furthermore based on the churn definition (no player activity during the CP period) given by Choi et al. (2017) each player was either labelled as a churner or not; from this exercise, it was noted that the dataset of the mobile game Dodge the Mud suffered from severe class imbalance.

Two experiments were devised to investigate data stationarity which involved using statistical tests such as the ADF and the decision tree complexity analysis. It was observed that stationarity was not that common in the features which reinforced the notion of concept drift. Thus, concept drift handling techniques were applied to the data, which involved designing two windowing approaches: the moving window and incremental window. Results from these approaches

did not show any significant improvement.

It was concluded that due to the class imbalance the machine learning model is unable to learn the concepts in the dataset and this could be seen in the confusion matrix results where the majority class was always the best predicted. Two conclusions can be drawn out of this investigation. First, applying concept drift techniques such as the moving window approach does not always result in significant performance boost as in the case of the first part of this study. Secondly, the importance of dataset ordering in a time series when examining churn prediction from data variability point of view. Dataset ordering makes it possible to apply meaningful analysis with regards to dataset complexity and data shifts.

The second part of this research involved collaborating with an industrial partner from the online iGaming industry. The industrial collaborator has a data science team which come up with reliable and realistic prediction models (churn amongst others). The CRM and other teams make use of these predictive models to retain and engage the customer as much as possible, ultimately increasing revenue.

The data science team was crucial in providing the knowledge of the best features to use for churn prediction and having prior knowledge and experience to churn prediction helped in the creation of the dataset that was used in this work. A time-specific field, namely the first deposit date was requested, in order to be able to order the time series correctly. This field was chosen since the first deposit date is the fulcrum in churn prediction. Since it is used to trigger the model to generate a prediction if a predefined number of days have elapsed from the first deposit date; thus acting as a trigger mechanism.

The lessons learned from the previous investigation on the work of Choi et al. (2017) were applied to the industrial partner's dataset. Initially, data stationarity tests were applied on the extensive feature set. Results show that the majority of features are non-stationary, which indicate that the dataset in hand suffers from concept drift. To further reinforce this outcome, a decision tree complexity analysis was carried out, where a parameter unbounded decision tree was fitted on the dataset. During each learning period, the node count and tree depth were noted together with pertinent dataset properties such as the churn count and churn ratio. This experiment showed that there were particular periods where

the node count spikes which indicates severe data shifts. Having dealt with concept drift analysis, two concept drift handling approaches being the moving window approach, and the window dissimilarity approach was applied on the dataset. The former consisted of sliding across the dataset at a specific stride wherein each window the model is trained on a specific number of rows and tested on several unseen rows in the future. The latter takes a similar approach but rather than creating a new model with every window; a dissimilarity metric is used to discern if the model needs to be updated or not. The dissimilarity metric was computed using the KS statistical test.

Results from the windowing techniques showed a statistically significant improvement when compared to the static model. Furthermore, a significant improvement was noted when compared to the results of the data science team, where an increase of around three per cent was noted. The LGBM model implemented by the industrial collaborator achieved a ROC-AUC metric of 0.739. In contrast, the moving window concept drift approach implemented in this study attained a ROC-AUC metric of 0.755. Additionally, the window dissimilarity approach also achieved similar results when compared to the moving window. However, in the window dissimilarity approach, the number of times when the model was updated resulted in being significantly smaller than that of the moving window model. In the moving window approach, the model was updated around 350 times while in the window dissimilarity approach around 70 times.

When applying this improvement for the number of players that are acquired daily results in a significant profit boost for the company. With the work carried on the industrial collaborator, this study achieved all the aims and objectives listed, and it could be concluded that albeit statistically significant improvement was attained on the collaborator data, the work on the research by Choi et al. (2017) did not show the same behaviour. Hence, cross-domain compatibility is not guaranteed. That is one needs to analyse each scenario on a case by case basis before implementing either of the concept drift approaches.

Through this discussion, the aims and objectives listed in the introductory chapter were attained, because dataset analysis in both parts of this work indicate that the data suffered from concept drift. Furthermore, several concept drift approaches were applied to both datasets. For the third objective, results show

that in the case of the dataset used in the work of Choi et al. (2017), no significant improvement was obtained. Contrastingly when applying these approaches to iGaming data, significant improvement was obtained. Thus one cannot assume cross-domain compatibility, and every case needs to be thoroughly evaluated for concept drift.

6.2 | Critique and Limitations

First, vanilla models were used in the second part of this study and no model hyper-parameter search was carried out, mainly due to two reasons. First, the dataset extracted from the industrial collaborator was large (over one million rows), and hence experiment duration was lengthy. Secondly, due to the lack of computing power available, model hyper-parameter optimisation was not an option. Although the dataset used in the work of Choi et al. (2017) did not contain a large volume of data, model hyper-parameter space search was not carried out, and this is a limitation of this study as with this approach results may possibly have been better.

Despite achieving statistically significant improvement when the concept drift approaches were applied to the industrial collaborator data, the two approaches were implemented on a limited data span of two and a half years. Albeit not a small range, it would have been better if the two concept drift approaches have been put into action and running in parallel with the production model of the industrial collaborator. By doing so, one can better understand and evaluate the benefit of such an approach in a real-life scenario.

6.3 | Future Work

This study suggests that concept drift techniques result in a performance boost, mainly when applied to iGaming data. Experiments were carried out using default model hyper-parameters, and hence model optimisation using random or grid search could provide potentially better results. Furthermore, as highlighted during the discussion of the limitations, this dataset contained categorical fea-

tures which were converted into distinct features using one-hot encoding. One-hot encoding induces data sparsity which can lead to performance degradation. In order to counteract this, a feature selection technique could be used so that during every window, the most important features are selected for model training. Feature selection could be implemented using recursive feature elimination technique amongst others. This suggestion is also tightly linked with the feature importance analysis carried in this study which highlighted that feature importance varies according to the window. Furthermore, dimensionality reduction techniques such as the Principle Component Analysis (PCA) can also be implemented as part of the feature optimisation.

As discussed in several sections throughout this work, in churn prediction scenario, it is all about being proactive. In doing so, a company will gain a competitive edge and thus increasing profit. In view of the work carried out, there are several improvements which can be made starting from the dataset features. This improvement, in particular, is concerned with the industrial collaborator dataset. When discussing the dataset creation steps with the data science team, they were asked if it was possible to extract information that describes the interaction of the customer with the customer service team through the use of the online messaging system. However, such integration was currently not possible. Features describing the customer service level satisfaction would be of a benefit to an early churn prediction model as the interaction during the early stages of customer acquisition is crucial. Essentially creating new possibilities in feature creation as natural language processing algorithms can be used to produce new diversified features such as online messaging system satisfaction score.

The area of churn prediction is tightly entwined with other areas in customer relationship management. Thus with improved churn prediction performance as achieved in this work, other areas within the business can be aided better. Customers that are predicted as highly likely to churn should be filtered further as it is implausible to retain all those customers. Customer filtering can be done by evaluating the Customer Lifetime Value (CLV) of those customers which were predicted as churner (according to the proposed concept drift framework of this study). Thus more effort is geared towards retaining the most profitable customers. Hence, providing a holistic approach to an effective customer re-

tention strategy as the customer data during the acquisition stage gives a good indication to the customer's profitability. Using this approach one will have two predictive models working in tandem. The first model will be the concept drift aware churn prediction model which was implemented in this study which triggers another proposed predictive model. The latter used to predict the customer's profitability and hence enhancing further the company's customer retention strategy.

Another future work which can be proposed is to implement the concept drift aware churn prediction framework in collaboration with the same partner so that it can be used as their production system. Being an online iGaming company, large volumes of transactional data need to be processed daily, and hence big data technologies are a must. In particular, Apache Spark, which is a big data distributed processing framework, could be used for this task. More so such framework contains all the necessary machine learning models used in this work, and hence real-time prediction is possible.

6.4 | Final Remarks

In today's growing online iGaming domain, companies strive to gain a competitive edge, and this is mainly carried out through the use of effective customer retention strategies. Inherently an effective customer retention strategy requires a proactive mentality which is made possible through the use of churn prediction techniques, where the company will have a good indication of which customers will churn. This research, which formed part of the industrial collaboration, tackled the problem of churn prediction by considering the effect of concept drift. Results showed that when implementing concept drift techniques performance is improved, which leads to a potential increase in customer retention and more significant profit margins.

A

Appendix

A.1 | Industrial Collaboration

Name	ADF	KPSS	Outcome
Referrer Btag	0.019	**0.010**	Non-stationary
Referrer Social	**0.135**	**0.010**	Non-stationary
Referrer Seo	**0.065**	**0.010**	Non-stationary
Referrer Article Free	**0.355**	**0.010**	Non-stationary
Referrer Mobile Url	**0.097**	**0.010**	Non-stationary
Referrer Domain Free	**0.155**	**0.010**	Non-stationary
Acquisition Affiliated	0.046	**0.010**	Non-stationary
Acquisition Direct	**0.432**	**0.023**	Non-stationary
Acquisition Organic	**0.139**	**0.010**	Non-stationary
Acquisition Other	0.002	**0.017**	Non-stationary
Gender Male	**0.088**	**0.010**	Non-stationary
Registration Birthday	0.000	0.080	Stationary
First Deposit Birthday	0.000	0.100	Stationary
Age Bucket 24	**0.058**	**0.010**	Non-stationary
Age Bucket 25 28	**0.071**	**0.010**	Non-stationary
Age Bucket 29 33	**0.059**	**0.010**	Non-stationary
Age Bucket 34 40	**0.148**	**0.010**	Non-stationary
Age Bucket 41 48	**0.187**	**0.010**	Non-stationary
Age Bucket 49	**0.412**	0.069	Non-stationary
Registration Night	0.046	**0.010**	Non-stationary

Continued on next page

Table A.1 – continued from previous page

Name	ADF	KPSS	Outcome
Registration Morning	**0.198**	**0.010**	Non-stationary
Registration Day1	**0.119**	**0.010**	Non-stationary
Registration Day2	**0.105**	**0.010**	Non-stationary
Registration Evening	**0.084**	**0.010**	Non-stationary
Same Day Conversion	0.000	**0.037**	Non-stationary
Player Phone Number Quality	**0.124**	**0.010**	Non-stationary
City Bucket	**0.094**	**0.010**	Non-stationary
Bet Mobile	0.000	**0.010**	Non-stationary
Bet Desktop	0.002	**0.010**	Non-stationary
Win Mobile	0.000	**0.010**	Non-stationary
Win Desktop	0.029	**0.010**	Non-stationary
Jackpot Contribution Mobile	0.021	**0.010**	Non-stationary
Jackpot Contribution Desktop	0.000	**0.010**	Non-stationary
Number of Rounds Mobile	**0.329**	0.100	Non-stationary
Number of Rounds Desktop	**0.106**	**0.010**	Non-stationary
Bet Bonus Mobile	**0.145**	**0.010**	Non-stationary
Bet Bonus Desktop	0.048	**0.010**	Non-stationary
Bet Locked Mobile	**0.287**	**0.010**	Non-stationary
Bet Locked Desktop	**0.539**	**0.010**	Non-stationary
Bet Real Mobile	0.000	**0.016**	Non-stationary
Bet Real Desktop	0.000	**0.010**	Non-stationary
Win Bonus Mobile	**0.128**	**0.011**	Non-stationary
Win Bonus Desktop	**0.055**	**0.010**	Non-stationary
Win Locked Mobile	**0.300**	**0.010**	Non-stationary
Win Locked Desktop	**0.521**	**0.010**	Non-stationary
Win Real Mobile	0.000	**0.016**	Non-stationary
Win Real Desktop	0.000	**0.010**	Non-stationary
Jackpot Contribution	0.000	**0.010**	Non-stationary
Jackpot Win	0.000	0.100	Stationary
Number of Rounds Slots	**0.336**	**0.010**	Non-stationary
Number of Rounds Black Jack	**0.074**	**0.010**	Non-stationary
Number of Rounds Table Game	0.000	**0.010**	Non-stationary
Number of Rounds Roulette	0.000	**0.010**	Non-stationary
Number of Rounds Lottery	**0.670**	**0.010**	Non-stationary
Number of Rounds Poker	0.000	**0.010**	Non-stationary

Continued on next page

Table A.1 – continued from previous page

Name	ADF	KPSS	Outcome
Number of Rounds Bingo Keno	0.000	**0.010**	Non-stationary
Number of Rounds Live Casino	0.038	**0.010**	Non-stationary
Number of Rounds Jackpot	0.001	**0.010**	Non-stationary
Number of Rounds Other	**0.470**	**0.010**	Non-stationary
Bet Slots	**0.092**	**0.010**	Non-stationary
Bet Black Jack	0.000	**0.010**	Non-stationary
Bet Table Game	0.000	0.100	Stationary
Bet Roulette	0.000	**0.010**	Non-stationary
Bet Lottery	0.010	**0.010**	Non-stationary
Bet Poker	0.000	**0.010**	Non-stationary
Bet Bingo Keno	0.000	**0.010**	Non-stationary
Bet Live Casino	0.000	**0.010**	Non-stationary
Bet Jackpot	0.000	**0.010**	Non-stationary
Bet Other	0.000	**0.010**	Non-stationary
Win Slots	**0.082**	**0.010**	Non-stationary
Win Black Jack	0.000	**0.010**	Non-stationary
Win Table Game	0.000	0.100	Stationary
Win Roulette	0.000	**0.010**	Non-stationary
Win Lottery	0.045	**0.010**	Non-stationary
Win Poker	0.000	**0.010**	Non-stationary
Win Bingo Keno	0.000	**0.012**	Non-stationary
Win Live Casino	0.000	**0.010**	Non-stationary
Win Jackpot	0.000	**0.010**	Non-stationary
Win Other	0.000	**0.010**	Non-stationary
Bet Real Slots	0.019	**0.010**	Non-stationary
Bet Real Black Jack	0.000	**0.010**	Non-stationary
Bet Real Table Game	0.000	0.100	Stationary
Bet Real Roulette	0.000	**0.010**	Non-stationary
Bet Real Lottery	0.002	**0.010**	Non-stationary
Bet Real Poker	0.000	**0.010**	Non-stationary
Bet Real Bingo Keno	0.000	0.067	Stationary
Bet Real Live Casino	0.000	**0.010**	Non-stationary
Bet Real Jackpot	0.000	**0.010**	Non-stationary
Bet Real Other	0.000	**0.010**	Non-stationary
Bet Bonus Slots	0.041	**0.010**	Non-stationary

Continued on next page

Table A.1 – continued from previous page

Name	ADF	KPSS	Outcome
Bet Bonus Black Jack	0.000	**0.010**	Non-stationary
Bet Bonus Table Game	**0.580**	**0.010**	Non-stationary
Bet Bonus Roulette	0.000	0.100	Stationary
Bet Bonus Lottery	**0.139**	**0.010**	Non-stationary
Bet Bonus Poker	0.000	**0.010**	Non-stationary
Bet Bonus Bingo Keno	0.000	**0.010**	Non-stationary
Bet Bonus Live Casino	**0.125**	**0.010**	Non-stationary
Bet Bonus Jackpot	0.000	**0.010**	Non-stationary
Bet Bonus Other	0.000	**0.010**	Non-stationary
Bet Locked Slots	**0.491**	**0.010**	Non-stationary
Bet Locked Black Jack	0.000	**0.021**	Non-stationary
Bet Locked Table Game	0.033	**0.010**	Non-stationary
Bet Locked Roulette	0.000	0.100	Stationary
Bet Locked Lottery	0.000	**0.010**	Non-stationary
Bet Locked Poker	0.000	0.100	Stationary
Bet Locked Bingo Keno	0.000	**0.031**	Non-stationary
Bet Locked Live Casino	**0.311**	**0.010**	Non-stationary
Bet Locked Jackpot	**0.181**	**0.010**	Non-stationary
Bet Locked Other	0.013	0.100	Stationary
Win Real Slots	0.015	**0.010**	Non-stationary
Win Real Black Jack	0.000	**0.010**	Non-stationary
Win Real Table Game	0.000	0.100	Stationary
Win Real Roulette	0.000	**0.010**	Non-stationary
Win Real Lottery	0.000	**0.010**	Non-stationary
Win Real Poker	0.000	**0.010**	Non-stationary
Win Real Bingo Keno	0.000	0.085	Stationary
Win Real Live Casino	0.000	**0.010**	Non-stationary
Win Real Jackpot	0.000	**0.010**	Non-stationary
Win Real Other	0.000	**0.010**	Non-stationary
Win Bonus Slots	0.041	**0.010**	Non-stationary
Win Bonus Black Jack	0.000	**0.010**	Non-stationary
Win Bonus Table Game	**0.418**	**0.010**	Non-stationary
Win Bonus Roulette	0.000	0.100	Stationary
Win Bonus Lottery	**0.553**	**0.010**	Non-stationary
Win Bonus Poker	0.000	**0.010**	Non-stationary

Continued on next page

Table A.1 – continued from previous page

Name	ADF	KPSS	Outcome
Win Bonus Bingo Keno	0.000	**0.010**	Non-stationary
Win Bonus Live Casino	**0.127**	**0.010**	Non-stationary
Win Bonus Jackpot	0.000	**0.016**	Non-stationary
Win Bonus Other	**0.426**	**0.010**	Non-stationary
Win Locked Slots	**0.616**	**0.010**	Non-stationary
Win Locked Black Jack	0.000	**0.021**	Non-stationary
Win Locked Table Game	0.019	**0.010**	Non-stationary
Win Locked Roulette	0.000	0.100	Stationary
Win Locked Lottery	0.000	**0.010**	Non-stationary
Win Locked Poker	0.000	0.100	Stationary
Win Locked Bingo Keno	0.000	0.079	Stationary
Win Locked Live Casino	**0.244**	**0.010**	Non-stationary
Win Locked Jackpot	**0.190**	**0.010**	Non-stationary
Win Locked Other	0.000	0.100	Stationary
Sum Deposits PSP	**0.065**	0.058	Non-stationary
Sum Deposits Bank Card	**0.250**	**0.010**	Non-stationary
Sum Withdrawals PSP	**0.184**	**0.010**	Non-stationary
Sum Withdrawals Bank Card	0.027	**0.010**	Non-stationary
Sum Deposits Failed PSP	**0.274**	**0.010**	Non-stationary
Sum Deposits Failed Bank Card	0.002	**0.010**	Non-stationary
Sum Withdrawals Failed PSP	0.000	**0.010**	Non-stationary
Sum Withdrawals Failed Bank Card	0.000	**0.010**	Non-stationary
Sum Withdrawals Cancelled PSP	0.000	**0.010**	Non-stationary
Sum Withdrawals Cancelled Bank Card	**0.190**	**0.010**	Non-stationary
Sum Withdrawals Reversed PSP	0.000	0.100	Stationary
Sum Withdrawals Reversed Bank Card	0.000	**0.010**	Non-stationary
Sum Withdrawals Rejected PSP	0.000	**0.010**	Non-stationary
Sum Withdrawals Rejected Bank Card	**0.129**	**0.010**	Non-stationary
Count Deposits PSP	**0.177**	0.100	Non-stationary
Count Deposits Bank Card	**0.436**	**0.010**	Non-stationary
Count Withdrawals PSP	**0.321**	**0.020**	Non-stationary
Count Withdrawals Bank Card	**0.211**	**0.010**	Non-stationary
Count Deposits Failed PSP	**0.777**	**0.010**	Non-stationary
Count Deposits Failed Bank Card	**0.159**	**0.010**	Non-stationary
Count Withdrawals Failed PSP	0.000	**0.010**	Non-stationary

Continued on next page

Table A.1 – continued from previous page

Name	ADF	KPSS	Outcome
Count Withdrawals Failed Bank Card	**0.067**	**0.010**	Non-stationary
Count Withdrawals Cancelled PSP	0.001	**0.010**	Non-stationary
Count Withdrawals Cancelled Bank Card	0.038	**0.010**	Non-stationary
Count Withdrawals Reversed PSP	0.000	0.100	Stationary
Count Withdrawals Reversed Bank Card	0.000	**0.010**	Non-stationary
Count Withdrawals Rejected PSP	**0.195**	**0.010**	Non-stationary
Count Withdrawals Rejected Bank Card	**0.288**	**0.010**	Non-stationary

Table A.1: P-values for ADF and KPSS Tests for all features.

References

Mohammad Saeed Abou Trab, Tariq Elyas, Mohammed Hassouna, and Ali Tarhini. Customer churn in mobile markets: A comparison of techniques. International Business Research, 8(6), May 2015. ISSN 1913-9004.

Sulaimon Olanrewaju Adebiyi, Bilqis Bolanle Amole, and Emmanuel Olateju Oyatoye. Relevant drivers for customers` churn and retention decision in the nigerian mobile telecommunication industry. Journal of Competitiveness, 6(3):52–67, September 2016.

Kang Adiwijaya, Abdurahman Baizal, and Veronikha Effendy. Handling imbalanced data in customer churn prediction using combined sampling and weighted random forest. In 2014 2nd International Conference on Information and Communication Technology (ICoICT). IEEE, May 2014.

Awais Adnan, Adnan Amin, Sajid Anwar, Ahmad Hawalah, Newton Howard, Amir Hussain, Muhammad Nawaz, and Junaid Qadir. Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study. IEEE Access, 4:7940–7957, 2016.

Afshin Afshari, Zeyar Aung, Nengbao Liu, and Wei Lee Woon. Handling class imbalance in customer behavior prediction. In 2014 International Conference on Collaboration Technologies and Systems (CTS), pages 100–103, 2014.

Sonali Agarwal, Pretam Jayaswal, Bakshi Rohit Prasad, and Divya Tomar. An ensemble approach for efficient churn prediction in telecom industry. International Journal of Database Theory and Application, 9(8):211–232, August 2016.

Ammara Ahmed and D. Maheswari Linen. A review and analysis of churn prediction methods for customer retention in telecom industries. In 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS). IEEE, January 2017.

Reem Al-Otaibi, Peter Flach, Meelis Kull, and Ricardo B. C. Prudêncio. Versatile decision trees for learning over multiple contexts. In Machine Learning and Knowledge Discovery in Databases, pages 184–199, Cham, 2015. Springer International Publishing. ISBN 978-3-319-23528-8.

Andry Alamsyah and Nisrina Salma. A Comparative Study Of Employee Churn Prediction Model. In 2018 4th International Conference on Science and Technology (ICST). IEEE, August 2018.

Matt Alhaery and Eunju Suh. Customer retention: Reducing online casino player churn through the application of predictive modeling. 2016.

Reda Alhajj, Keivan Kianmehr, Mick J. Ridley, and Mohammad Rifaie. Data warehouse architecture and design. In 2008 IEEE International Conference on Information Reuse and Integration, pages 58–63, 2008.

Cesare Alippi, Gianluca Bontempi, Giacomo Boracchi, Olivier Caelen, and Andrea Dal Pozzolo. Credit card fraud detection and concept-drift adaptation with delayed supervised information. In 2015 International Joint Conference on Neural Networks (IJCNN). IEEE, July 2015.

Tor Wallin Andreassen, Jaesung Cha, Anders Gustafsson, Michael D. Johnson, and Line Lervik. The evolution and future of national customer satisfaction index models. Journal of Economic Psychology, 22(2):217–245, April 2001.

Jirawat Anuwichanont. The impact of price perception on customer loyalty in the airline context. Journal of Business & Economics Research (JBER), 9(9):37, August 2011.

Seyed M.S. Ardabili and Abbas Keramati. Churn analysis for an iranian mobile operator. Telecommunications Policy, 35(4):344–356, May 2011.

Noor Azhar and Aamer Hanif. Resolving class imbalance and feature selection in customer churn dataset. In 2017 International Conference on Frontiers of Information Technology (FIT). IEEE, December 2017.

Yair Babad, Chandrasekaran Ranganathan, and DongBack Seo. Two-level model of customer retention in the US mobile telecommunications service market. Telecommunications Policy, 32(3-4):182–196, April 2008.

Bart Baesens, Jochen De Weerdt, Wilfried Lemahieu, and Sandra Mitrović. On the operational efficiency of different feature types for telco churn prediction. European Journal of Operational Research, 267(3):1141–1155, June 2018.

Yang Bai, Honghua Dai, Saeid Nahavandi, Yange Sun, and Zhihai Wang. A classifier graph based recurring concept detection and prediction approach. Computational Intelligence and Neuroscience, 2018:1–13, June 2018.

N. Balakrishnan, Norman L. Johnson, Samuel Kotz, Campbell B. Read, and Brani Vidakovic. Encyclopedia of Statistical Sciences. Wiley Online Library, July 2004.

Aditya Bankar, Preeti K. Dalvi, Ashish Deomore, Vijay A. Kanade, and Siddhi K. Khandge. Analysis of customer churn prediction in telecom industry using decision trees and logistic regression. In 2016 Symposium on Colossal Data Analysis and Networking (CDAN). IEEE, March 2016.

Aristide Baratin, Simon Lacoste-Julien, Loannis Mitliagkas, Sarthak Mittal, Brady Neal, Matthew Scicluna, and Vinayak Tantia. A modern take on the bias-variance tradeoff in neural networks. ArXiv, abs/1810.08591, 2018.

BBC. Machine learning history. Online, 2019. URL `https://www.bbc.com/timelines/zypd97h`. Last Accessed on 07-09-2020.

George Bebis, Ronald Miller, and Zehang Sun. Object detection using feature subset selection. Pattern Recognition, 37(11):2165–2176, November 2004.

Neeli Bendapudi and Leonard L. Berry. Customers' motivations for maintaining relationships with service providers. Journal of Retailing, 73(1):15–37, March 1997.

Shrisha Bharadwaj, Anil B.S., P S Gowra, Sharath Kumar, Abhiraj Pahargarh, and Adhiraj Pahargarh. Customer churn prediction in mobile networks using logistic regression and multilayer perceptron(MLP). In 2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT). IEEE, August 2018.

Albert Bifet, Abdelhamid Bouchachia, João Gama, Mykola Pechenizkiy, and Indrė Žliobaitė. A survey on concept drift adaptation. ACM Computing Surveys, 46(4):1–37, March 2014.

Tammo H. A. Bijmolt, Hans Risselada, and Peter C. Verhoef. Staying Power of Churn Prediction Models. Journal of Interactive Marketing, 24(3):198–208, August 2010.

Luo Bin, Liu Juan, and Shao Peiji. Customer churn prediction based on the decision tree in personal handyphone system service. In 2007 International Conference on Service Systems and Service Management. IEEE, June 2007.

Koen W. De Bock, Arno De Caigny, and Kristof Coussement. A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. European Journal of Operational Research, 269(2):760–772, September 2018.

Leo Breiman. Random forests. Machine Learning, 45(1):5–32, 2001.

Jason Brownlee. Supervised unsupervised machine learning, March 2016. URL https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/. Last Accessed on 02-05-2020.

Manfred Bruhn and Michael A. Grund. Theory, development and implementation of national customer satisfaction indices: The swiss index of customer satisfaction (SWICS). Total Quality Management, 11(7):1017–1028, September 2000.

Wouter Buckinx and Dirk Van den Poel. Customer base analysis: partial defection of behaviourally loyal clients in a non-contractual FMCG retail setting. European Journal of Operational Research, 164(1):252–268, July 2005.

Jonathan Burez and Dirk Van den Poel. CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services. Expert Systems with Applications, 32(2):277–288, February 2007.

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(1-2):497–514, July 2008.

Jonathan Burez and Dirk Van den Poel. Handling class imbalance in customer churn prediction. Expert Systems with Applications, 36(3):4626–4636, April 2009.

Francis Buttle. Customer Relationship Management. Routledge, February 2004.

Hakki Candan Cankaya, Turker Ince, and Utku Yabas. Customer Churn Prediction for Telecom Services. In 2012 IEEE 36th Annual Computer Software and Applications Conference. IEEE, July 2012.

Marius Capota and Vladislav Lazarov. Churn prediction. Business Analytics Course. TUM Computer Science, 2007.

Gladys Castillo, João Gama, Pedro Medas, and Pedro Rodrigues. Learning with drift detection. In Learning with Drift Detection, volume 8, pages 286–295, September 2004.

Hsu-Hwa Chang, Mu-Chen Chen, and Ai-Lun Chiu. Mining changes in customer behavior in retail marketing. Expert Systems with Applications, 28(4):773–781, May 2005.

Dorothy C. K. Chau, Eric W. T. Ngai, and Li Xiu. Application of data mining techniques in customer relationship management: A literature review and classification. Expert Systems with Applications, 36(2):2592–2602, March 2009.

Tianqi Chen and Carlos Guestrin. Xgboost. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, August 2016.

Wei Chen, Thomas Finley, Guolin Ke, Tie-Yan Liu, Weidong Ma, Qi Meng, Taifeng Wang, and Qiwei Ye. Lightgbm: A highly efficient gradient boosting decision tree. In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, page 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Daeyoung Choi, Seungwook Kim, Eunjung Lee, and Wonjong Rhee. Churn prediction of mobile and online casual games using play log data. PLOS ONE, 12(7):e0180735, July 2017.

Bong-Horng Chu, Cheng-Seen Ho, and Ming-Shian Tsai. Toward a hybrid data mining model for customer retention. Knowledge-Based Systems, 20(8):703–718, December 2007.

Michael D. Clemes, Christopher Gan, and Dongmei Zhang. Customer switching behaviour in the chinese retail banking industry. International Journal of Bank Marketing, 28(7):519–546, October 2010.

Kristof Coussement and Koen W. De Bock. Customer churn prediction in the online gambling industry: The beneficial effect of ensemble learning. Journal of Business Research, 66(9):1629 – 1636, 2013. Advancing Research Methods in Marketing.

Shaoying Cui and Ning Ding. Customer churn prediction using improved FCM algorithm. In 2017 3rd International Conference on Information Management (ICIM). IEEE, April 2017.

Kiran Dahiya and Kanika Talwar. Customer churn prediction in telecommunication industries using data mining techniques- a review. International Journal of Advanced Research in Computer Science and Software Engineering, 5(4):417–43, April 2015.

Reham Dannoun, Hossam Faris, Osama Harfoushi, Amjad Hudaib, and Ruba Obiedat. Hybrid data mining models for predicting customer churn. International Journal of Communications, Network and System Sciences, 08(05):91–96, 2015.

Natalie Davies. The evolution of crm - the path to a happy customer, 2013. URL `https://www.sales-i.com/the-evolution-of-crm-the-path-to-a-happy-customer.` Last Accessed on 08-11-2019.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In Proceedings of the 23rd International Conference on Machine Learning, ICML '06, page 233–240, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832.

John Dawson and Chieko Minami. The CRM process in retail and service sector firms in japan: Loyalty development and financial return. Journal of Retailing and Consumer Services, 15(5): 375–385, September 2008.

M. P. Deepshika, Samarth S. Hiremath, Shantam Mittal, Annapurna P. Patil, Yogesh E. Patil, and Savita Shetty. Customer churn prediction for retail business. In 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS). IEEE, August 2017.

Duyen Do, Phuc Huynh, Phuong Vo, and Tu Vu. Customer churn prediction in an internet service provider. In 2017 IEEE International Conference on Big Data (Big Data), pages 3928–3933, 2017.

Duyen Do, Phuc Huynh, Phuong Vo, and Tu Vu. Customer churn prediction in an internet service provider. In 2017 IEEE International Conference on Big Data (Big Data). IEEE, December 2017.

Pedro Domingos. A few useful things to know about machine learning. Communications of the ACM, 55(10):78, October 2012.

Fan Dong, João Gama, Feng Gu, Anjin Liu, Jie Lu, and Guangquan Zhang. Learning under concept drift: A review. IEEE Transactions on Knowledge and Data Engineering, 31(12): 2346–2363, 2019.

Shoubin Dong, Jinlong Hu, Minjie Huang, Lei Lei, Jiang Yang, Runchao Zhu, and Yi Zhuang. pRNN: A recurrent neural network based approach for customer churn prediction in telecommunication sector. In 2018 IEEE International Conference on Big Data (Big Data). IEEE, December 2018.

Bas Donkers and Peter C. Verhoef. Predicting customer potential value an application in the insurance industry. Decision Support Systems, 32(2):189–199, December 2001.

Kent Eriksson and Anna Lofmarck Vaghult. Customer retention, purchasing behavior and relationship substance in professional services. Industrial Marketing Management, 29(4):363–372, July 2000.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Extremely randomized trees. Machine Learning, 63(1):3–42, March 2006.

Abdolreza Eshghi, Dominique Haughton, and Heikki Topi. Determinants of customer loyalty in the wireless telecommunications industry. Telecommunications Policy, 31(2):93–106, March 2007.

Boi Faltings, Florent Garcin, Pierangelo Rothenbuehler, and Julian Runge. Hidden markov models for churn prediction. In 2015 SAI Intelligent Systems Conference (IntelliSys). IEEE, November 2015.

Pablo Farías, Pedro Hidalgo, Enrique Manzur, and Sergio Olavarrieta. Customer retention and price matching: The AFPs case. Journal of Business Research, 61(6):691–696, June 2008.

Peter Flach. Performance evaluation in machine learning: The good, the bad, the ugly and the way forward. In 33rd AAAI Conference on Artificial Intelligence, 2019.

Alvis Fong, Affan Ahmed Toor, Muhammad Usman, and Farah Younas. A robust systematic approach for ensuring optimal telecom service delivery. IEEE Communications Magazine, 58 (8):49–53, 2020.

Alan Fontaine. Mastering Predictive Analytics with scikit-learn and TensorFlow. Packt Publishing, 2018. ISBN 178961774X. URL https://www.ebook.de/de/product/34496587/alan_fontaine_mastering_predictive_analytics_with_scikit_learn_and_tensorflow.html. Last Accessed on 01-02-2020.

I. Franciska and B. Swaminathan. Churn prediction analysis using various clustering algorithms in KNIME analytics platform. In 2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS). IEEE, May 2017.

Amy Gallo. The value of keeping the right customers, October 2014. URL https://hbr.org/2014/10/the-value-of-keeping-the-right-customers. Last Accessed on 20-10-2019.

João Gama, Pedro Pereira Rodrigues, and Raquel Sebastião. Evaluating algorithms that learn from data streams. In Proceedings of the 2009 ACM Symposium on Applied Computing, SAC '09, page 1496–1500, New York, NY, USA, 2009a. Association for Computing Machinery. ISBN 9781605581668.

João Gama, Pedro Pereira Rodrigues, and Raquel Sebastião. Issues in evaluation of stream learning algorithms. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09, page 329–338, New York, NY, USA, 2009b. Association for Computing Machinery. ISBN 9781605584959.

Joshua S. Gans. Network competition and consumer churn. Information Economics and Policy, 12(2):97–109, June 2000.

Pierre Geurts and Gilles Louppe. Learning to rank with extremely randomized trees. In Proceedings of the 2010 International Conference on Yahoo! Learning to Rank Challenge - Volume 14, YLRC'10, page 49–61. JMLR.org, 2010.

Hajar Ghaneei, Abbas Keramati, and Seyed Mohammad Mirmohammadi. Developing a prediction model for customer churn from electronic banking services using data mining. Financial Innovation, 2(1), August 2016.

Mohammad Reza Gholamian, Sahand Khakabimamaghani, and Morteza Namvar. Data mining applications in customer churn management. In 2010 International Conference on Intelligent Systems, Modelling and Simulation. IEEE, January 2010.

Sujoy Ghose, Mamata Jenamani, and Pratap K.J. Mohapatra. A stochastic model of e-customer behavior. Electronic Commerce Research and Applications, 2(1):81–94, March 2003.

Anjana Gosain and Saanchi Sardana. Handling class imbalance problem using oversampling techniques: A review. In 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pages 79–85, 2017.

Liz Grech. SiGMA and Malta's rise in the Online Gaming Industry. Malta Insideout, May 2016. URL `http://maltainsideout.com/25832/sigma-maltas-rise-online-gaming-industry/`. Last Accessed on 29-04-2020.

Bryan Gregory. Predicting customer churn: Extreme gradient boosting with temporal data. arXiv, 2018.

Prince Grover. Gradient boosting from scratch, 2017. URL `https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d`. Last Accessed on 16-02-2020.

Roya Hejazinia and Mahdi Kazemi. Prioritizing factors influencing customer churn. In Interdisciplinary Journal of Contemporary Research in Business, 2014.

Ed Herranz. Unit root tests. Wiley Interdisciplinary Reviews: Computational Statistics, 9:e1396, March 2017.

Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & Knowledge Management Process, 5 (2):01–11, March 2015.

Bin Hou, Min Li, and Tao Li. A study on the relation among customer perceived value, customer satisfaction and customer value of service enterprises. In 2011 International Conference on Computer Science and Service System (CSSS). IEEE, June 2011.

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(2):181–188, February 2004.

Muhammad Imran, Saif Ul Islam, Sung Won Kim, Ahmad Kamran Malik, Basit Raza, and Irfan Ullah. A churn prediction model using random forest: Analysis of machine learning techniques for churn prediction and factor identification in telecom sector. IEEE Access, 7: 60134–60149, 2019.

Gareth M. James, Trevor Hastie, Robert Tibshirani, and Daniela Witten. An Introduction to Statistical Learning. Springer-Verlag GmbH, 2017. ISBN 1461471370. URL https://www.ebook.de/de/product/20292548/gareth_james_daniela_witten_trevor_hastie_robert_tibshirani_an_introduction_to_statistical_learning.html. Last Accessed on 24-06-2020.

Dong-Heon Jeong, Moon-Koo Kim, and Myeong-Cheol Park. The effects of customer satisfaction and switching barrier on customer loyalty in korean mobile telecommunication services. Telecommunications Policy, 28(2):145–159, March 2004.

Pingjun Jiang and Bert Rosenbloom. Customer intention to return online: price perception, attribute-level performance, and satisfaction unfolding over time. European Journal of Marketing, 39(1/2):150–174, January 2005.

Mohammad Reza Keyvanpour and Soheila Mehr Molaei. An analytical review for event prediction system on time series. In 2015 2nd International Conference on Pattern Recognition and Image Analysis (IPRIA). IEEE, March 2015.

Shahzad Ali Khan and Zeeshan Ali Rana. Evaluating performance of software defect prediction models using area under precision-recall curve (auc-pr). In 2019 2nd International Conference on Advancements in Computational Sciences (ICACS), pages 1–6, 2019.

Will Koehrsen. An implementation and explanation of the random forest in python, 2018. URL https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76. Last Accessed on 04-05-2019.

Igor Kononenko and Matjaž Kukar. Chapter 3 - machine learning basics. In Igor Kononenko and Matjaž Kukar, editors, Machine Learning and Data Mining, pages 59 – 105. Woodhead Publishing, 2007. ISBN 978-1-904275-21-3.

Alexander H. Kracklauer, Daniel Quinn Mills, and Dirk Seifert, editors. Collaborative Customer Relationship Management. Springer Berlin Heidelberg, 2004.

Bart Larivière and Dirk Van den Poel. Investigating the role of product features in preventing customer churn, by using survival analysis and choice modeling: The case of financial services. Expert Systems with Applications, 27(2):277–285, August 2004.

Jie Lin and Xu Xu. A novel approach to discovering patterns of global customer retention. In 2008 IEEE Symposium on Advanced Management of Information for Globalized Enterprises (AMIGE). IEEE, September 2008.

Raymond Ling and David C. Yen. Customer Relationship Management: An Analysis Framework and Implementation Strategies. Journal of Computer Information Systems, 41(3):82–97, 2001.

Anjin Liu, Jie Lu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under Concept Drift: A Review. IEEE Transactions on Knowledge and Data Engineering, pages 1–1, 2018.

Duen-Ren Liu and Ya-Yueh Shih. Integrating AHP and data mining for product recommendation based on customer lifetime value. Information & Management, 42(3):387–400, March 2005.

Huan Liu and KianSing Ng. Customer Retention via Data Mining. Artificial Intelligence Review, 14(6):569–590, 2000.

Yong Liu and Yongrui Zhuang. Research model of churn prediction based on customer segmentation and misclassification cost in the context of big data. Journal of Computer and Communications, 03(06):87–93, 2015.

Yuxi (Hayden) Liu. Python Machine Learning By Example. Packt Publishing, 2017. ISBN 1783553111. URL `https://www.ebook.de/de/product/29461016/yuxi_hayden_liu_python_machine_learning_by_example.html`. Last Accessed on 09-08-2020.

Feng Long, Qi Tang, Guoen Xia, and Xianquan Zhang. A customer churn prediction model based on xgboost and mlp. In 2020 International Conference on Computer Engineering and Application (ICCEA), pages 608–612, 2020.

Nielsen L. R. Machado and Duncan D. A. Ruiz. Customer: A novel customer churn prediction method based on mobile application usage. In 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC), pages 2146–2151, 2017.

Renuka Mahajan, Vishal Mahajan, and Richa Misra. Review of data mining techniques for churn prediction in telecom. Journal of Information and Organizational Sciences, 39:183–197, December 2015.

Jan Mandák. Proposal and Implementation of Churn Prediction System for Telecommunications Company. PhD thesis, VŠB - Technical University of Ostrava, 2018.

Katy Micallef. Disrupting the iGaming industry. SiGMA, October 2019. URL `https://www.sigma.com.mt/news/disrupting-the-igaming-industry`. Last Accessed on 29-04-2020.

Mary Walowe Mwadulo. A review on feature selection methods for classification tasks. International Journal of Computer Applications Technology and Research, 5(6):395 – 402, 2016.

Nadeem Ahmad Naz, Shahzad M. Sarfraz, and Umar Shoaib. A review on customer churn prediction data mining modeling techniques. Indian Journal of Science and Technology, 11 (27):1–7, July 2018.

Ananthanarayanan Parasuraman. Customer service in business-to-business markets: an agenda for research. Journal of Business & Industrial Marketing, 13(4/5):309–321, August 1998.

Parag C. Pendharkar. Genetic algorithm based neural network approaches for predicting churn in cellular wireless network services. Expert Systems with Applications, 36(3):6714–6720, April 2009.

Zhicheng Qin. The factors influencing low-cost airline passenger satisfaction and loyalty in bangkok, thailand. Master's thesis, University of the Thai Chamber of Commerce, 2012.

Marc Rehmsmeier and Takaya Saito. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3):1–21, March 2015.

Frederick F. Reichheld and Earl W. Sasser. Zero defections: Quality comes to services. Harvard Business Review, 1990. URL `https://hbr.org/1990/09/zero-defections-quality-comes-to-services`. Last Accessed on 27-12-2020.

Sharyn Rundle-Thiele. Elaborating customer loyalty: exploring loyalty to wine retailers. Journal of Retailing and Consumer Services, 12(5):333–344, September 2005.

Hamed Shourabizadeh and Ibrahim Onuralp Yigit. An approach for predicting employee churn by using data mining. In 2017 International Artificial Intelligence and Data Processing Symposium (IDAP). IEEE, September 2017.

Sanyam Shukla and Sanjay Yadav. Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In 2016 IEEE 6th International Conference on Advanced Computing (IACC), pages 78–83, 2016.

Joffre Swait and Jill Sweeney. The effects of brand credibility on customer loyalty. Journal of Retailing and Consumer Services, 15(3):179–193, May 2008.

Thomas Teal. Loyalty Effect. Harvard Business Review Press, 2001. ISBN 1578516870. URL `https://www.ebook.de/de/product/3249504/thomas_teal_loyalty_effect.html`. Last Accessed on 01-09-2020.

Russell S. Winer. A framework for customer relationship management. California Management Review, 43(4):89–105, July 2001.

Staff Writer. A brief history of customer relationship management | crm switch. Online, 2013. URL https://www.crmswitch.com/crm-industry/crm-industry-history/. Last Accessed on 09-11-2019.

Ming-hai Ye and Qing-hua Zhai. An empirical study of the effect of customer satisfaction and its two Dimensions on Online Customer Loyalty. In 2009 IEEE International Conference on Industrial Engineering and Engineering Management. IEEE, December 2009.

Alice Zheng, Nicole Shelby, and Ellie Volckhausen. Evaluating machine learning models. O'Reilly Media Inc., 2015.

Xiuxia Zuo. Several important unit root tests. In 2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP), pages 10–14, 2019.

Indrė Žliobaitė. Learning under concept drift: An overview. Faculty of Mathematics and Informatics, October 2010.