DEGREE PROJECT IN INFORMATION AND COMMUNICATION TECHNOLOGY,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2017

Churn Analysis in a Music Streaming Service

Predicting and understanding retention

GUILHERME DINIS CHALIANE JUNIOR

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

KTH Information and
Communication Technology

Churn Analysis in a Music Streaming Service

Predicting and understanding retention

GUILHERME DINIS CHALIANE JUNIOR

Master's Thesis at KTH Information and Communication Technology
Supervisor: Vladimir Vlassov
Examiner: Sarunas Girdzijauskas

Trita-ICT-EX-2017:158

Abstract

Churn analysis can be understood as a problem of predicting and understanding abandonment of use of a product or service. Different industries ranging from entertainment to financial investment, and cloud providers make use of digital platforms where their users access their product offerings. Usage often leads to behavioural trails being left behind. These trails can then be mined to understand them better, improve the product or service, and to predict churn. In this thesis, we perform churn analysis on a real-life data set from a music streaming service, Spotify AB, with different signals, ranging from activity, to financial, temporal, and performance indicators. We compare logistic regression, random forest, along with neural networks for the task of churn prediction, and in addition to that, a fourth approach combining random forests with neural networks is proposed and evaluated. Then, a meta-heuristic technique is applied over the data set to extract Association Rules that describe quantified relationships between predictors and churn. We relate these findings to observed patterns in aggregate level data, finding probable explanations to how specific product features and user behaviours lead to churn or activation. For churn prediction, we found that all three non-linear methods performed better than logistic regression, suggesting the limitation of linear models for our use case. Our proposed enhanced random forest model performed mildly better than conventional random forest.

Referat

Churn Analys för en Musikstreamingtjänst: prediktera och förstå bibehållande av användare

Churn analys kan förstås som ett tillvägagångssätt för att prediktera och förstå avslutad användning av en produkt eller tjänst. Olika industrier, som kan sträcka sig från underhållning till finansiell investering och molntjänsteleverantörer, använder digitala plattformar där deras användare har tillgång till deras produkter. Användning leder ofta till efterlämnande av beteendemönster. Dessa beteendemönster kan därefter utvinnas för att bättre förstå användarna, förbättra produkterna eller tjänsterna och för att prediktera churn. I detta arbete utför vi churn analys på ett dataset från en musikstreamingtjänst, Spotify AB, med olika signaler, som sträcker sig från aktivitet, till finansiella och temporala samt indikationer på prestanda. Vi jämför logistisk regression, random forest och neurala nätverk med uppgiften att utföra churn prediktering. Ytterligare ett tillvägagångssätt som kombinerar random forests med med neurala nätverk föreslås och utvärderas. Sedan, för att ta fram regler som är begripliga för beslutstagare, används en metaheuristisk teknik för datasetet, som beskriver kvantifierade relationer mellan prediktorer och churn. Vi sätter resultaten i relation till observerade mönster hos aggregerad data, vilket gör att vi hittar troliga förklaringar till hur specifika karaktärer hos produkten och användarmönster leder till churn. För prediktering av churn gav samtliga icke-linjära metoder bättre prestanda än logistisk regression, vilket tyder på begränsningarna hos linjära modeller för vårt användningsfall, och vår föreslagna förbättrade random forest modell hade svagt bättre prestanda än den konventionella random forest.

Acknowledgments

"What is now proved was once only imagined.", William Blake

First of all, I would like to express my deepest gratitude to my industrial and academic supervisors, Edvard Wendelin and professor Vladimir Vlassov, respectively. Their guidance and assistance proved invaluable to the inception and completion of this work.

I would also like to thank my colleagues at Spotify AB whom, on repeated occasions, made themselves available for valuable discourse. In particular, I'd like to extend a special thank you my team: Andressa, Artem, Luka, Martin, Matt, Mushfiq, and Yuri; their support and companionship was nothing short of memorable. And also, Magnus and Thuy for their critical ears.

Finally, I would like to thank professor Sarunas Girdzijauskas, for being my examiner. To my classmates, Filip and Philipp, I am grateful for their continuous reviews and discussions, and to Maja for her invaluable time. Last, but not least, I'd like to extend my gratitude to my family, on whom I can always count on for support.

Stockholm, September 3, 2017
Guilherme Dinis Chaliane Junior

Contents

Chapter 1

Introduction

In the era of multi-device experiences, service providers with digital channels rely on the Internet as the single or main medium of delivery of their offerings to end users. Customers, then, engage with a platform with the purpose of achieving a given task, e.g. transferring money, teleconferencing, or streaming a film. For a business, this presents both a challenge and an opportunity.

The challenge is that one can no longer make assumptions about the given context in which a user will consume a service. Take listening to music, for instance. People do so while commuting to work, during physical workout, while driving, or in their homes. At each instance, they may go to different platforms for their listening experience [1, 2]. For an organization operating on a global scale, cultural nuances may still affect how aesthetics and quality are judged by their users [3]. Hence, when designing for an engaging user experience, the need arises to understand the different contexts in which their offerings are being consumed by users. One way to do this is to break the signals from users into distinct groups, and study how they relate to engagement, as attempted by Lehmann et al. [4]. With this challenge, and consequent understanding, comes the opportunity for growth, by catering to different types of users.

The reason why churn is important is because it affects a company's profit directly [5]. For every customer, there is an acquisition cost involved. Once they become an active user, this cost can be offset after a given period of time, depending on the businesses' formulation of their customers' life time value (LTV) [6]. But, if a customer leaves before reaching this point, then the business suffers a long term loss, along with the sunk cost that went into acquiring the customer. This applies to financial services, like banking, and investment; telecommunications services, such like broadband providers; and entertainment businesses like Netflix and Apple Music.

Despite the differences in their products, these services have several factors in common, from a problem-domain point of view. First, what users do on their platforms can be translated into events, with metadata, and event specific information about the experience or process. For example, if they listen to music, what type of songs they listen to, and for how long; or similarly, if they make an investment, what kind of business do they invest in, and how much of their savings do they invest. Second, the observed behaviour of users, as it was just described, can change over time. The music a person listens to and their capacity for risk taking can vary by season, and life stage [7, 8, 9, 10]. And third, most of these services are somewhat commoditized, so users have alternatives to switch to and, therefore, businesses must compete to improve their offerings.

In this study, we aim to look at churn in a specific context, from the predictive and descriptive dimensions. First, we engineer features to describe users' behaviour over a time window of their first 7 days of activity in a music streaming service, Spotify AB. Following, we used these features to predict if they remain with the service the subsequent week. Then, to understand how user actions and product traits affect churn, we perform Quantitative Association Rule Mining over the data to find rules explaining the outcome of retention with regards to the signals we engineered. The details of this analysis is describe in the chapters to come. For now, we start by describing our problem formulation, purpose, and goals.

1.1 Problem Formulation

At Spotify AB, there are different types of user subscription packages. As a freemium service, there are free packages with limited features and paid packages that get extra service features. Users can sign up, and change plans whenever they wish to. For our study, we considered any newly registered user, provided that they were on a free plan for at least one day. This includes users that registered to a free plan and later converted to a paid one, as well as users that registered for a paid plan and later converted to a free one.

When a user registers, they can begin streaming content immediately from one of the platforms the service is available through: mobile, desktop, home devices, game consoles, etc. We are interested in predicting if a newly registered user will be activate in the second week following their registration (see Figure 1.1 on the facing page). To do this, we take data gathered from the first week, called the observation window, and use it to train models to perform the prediction. To classify users as activating or churning, we simply check to see if there is any streaming activity from that user on the second week. If so, we consider the user as activating, and otherwise, churning.

The reasons for keeping the observation and activation windows relatively small is motivated by internal prior studies on the same population of users which indicated high churn probability two weeks after registration. In order to prevent this from happening, we need to anticipate churners before they leave, so we try to predict their likelihood of retention from their first week's experience. In addition to this, we also want to understand how certain aspects of the user experience, as measured by user's activity as well as other aspects like demographics, financial data (e.g. plan subscription), and performance (e.g. stream latency) during the

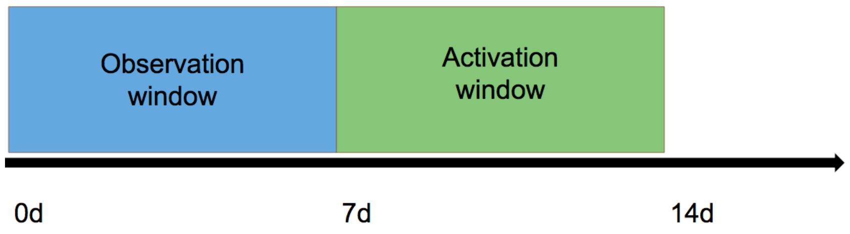


Figure 1.1: Observation and Activation Windows (in days)

first days of use affect retention.

Finally, the motivations for ensuring that all sampled users had been on a free plan for at least one day are two fold. First, all paid long term plans last at least one month, and users that have paid are more likely to continue using the service afterwards. And second, despite the fact they do not contribute through direct payment to the company's revenue, there is still value in free users as they add to the growth of the service as a platform, and they may later convert to a paid subscription.

1.2 Purpose

The purpose of this study is to (1) enable businesses to gain an understanding of what drives users to churn and remain, by analyzing activity, financial, demographic, and performance data to formulate hypotheses, and (2) get higher prediction performance in detecting potentially churning customers, thus reducing the costs of retention marketing campaigns. This study is conducted on real-world data, from a music streaming service.

1.3 Goal

We aim to better understand users and churn by leveraging log data. To achieve this, two analyses studies will be conducted. First, for churn prediction, three distinct methods are tested independently. Specifically, Logistic Regression, Random Forest, and Artificial Neural Network are compared. We propose an alternate technique to enhance the performance of an ensemble of Decision Trees, and evaluate it against Random Forest and Artificial Neural Network. Second, to identify relationships between features and activation, the technique of Quantitative Association Rule Mining is applied over the data set.

1.4 Hypotheses

Having defined our problem, and from our review of prior work, combined with earlier studies of the population in question, we arrive at the following hypotheses:

1. Non-activity related aspects like performance, demographics, and financial data are informative in churn prediction, as measured by impact to F Measure (F1-score) in a Random Forest (RF) model;

2. The proposed method for augmenting ensembles should yield higher F1-score in a RF.

1.5 Benefits, Ethics, and Sustainability

1.5.1 Benefits

As a comprehensive study on signals from user activity and their relation to churn, this research has obvious benefits to organizations with digital platforms: (1) identifying churning customers enables them to take preventive action and (2) understanding what drives churn helps them build better products for their customers and consumers. To customers and consumers, they get the benefit of having a better product as an alternative; one that addresses their needs and justifies their loyalty.

1.5.2 Ethics

Despite having ethical standards in place, several instances of privacy violation and lack of upfront disclosure can be encountered in digital technology firms. As an illustration, the rules and regulations from social media entities essentially require users to agree to participate in any present and future research projects that will involve their data, with or without intervention from the companies or user, as a prerequisite for them to make use of their services without a financial charge [11]. In most cases, the option to pay a fee and opt out of such terms is not made available. Simply put, users are placed in a position where they are agreeing to participate in studies that have not even been conceived, and therefore cannot possibly have a complete understanding of to give full consent [12]. This is why bodies such as the European Data Privacy Directive try to outline clear directives of access to individuals' data for said institutions, and settle any conflicts when they arise.

In 2015 the European Commission approved a new comprehensive, and updated regulation on the protection of personal information [13]. Several companies in Europe already follow guidelines on how long they may keep personally identifiable information; but the new guidelines also stipulate what identifiable information may be collected, and for what reasons, and it gives users the right to have their data erased. The data used for this study follows these guidelines, both in its requirements of non-identifiability of users and data retention.

For organizations that try to understand their users, most analysis is done on aggregate behaviour. Nonetheless, despite being performed on aggregate data, the findings of these studies can be used to, correctly or incorrectly, extrapolate behavioural traits of a somewhat personal nature. We cannot disregard the interest of the public, as measured by benefits and potential harms. Behavioural studies, when carried out in Social Sciences, are intended to enhance our understanding of how we work as individuals, and as a society. Due to their nature, they are guided by ethical principles of the institutions in which they are carried out, or by a larger body of the field. This study operates in a separate field, but encompasses the same parameters: people and data;

Pondering on harm, for instance, a study such as this could be used to favor certain users over others, by narrowing resources of basic product usage to particular user groups, creating unfairness. If that kind of practice were to be spread in society, it would create disparity in access to goods and services, which in our case would be mainly music. Further, studies on user's behaviour and preferences can easily be used for non-ethical means of coercion, e.g. asking a user to pay for the service when they are about to listen to their favorite music at their favorite time. To safeguard against this, we referred to standards like the Association Of Computing Machinery (ACM) *Software Engineering Code of Ethics and Professional Practice*, taking measures to safeguard the protection of the data used, results published, and its application.

1.5.3 Sustainability

A discussion of sustainability in our context requires us to turn our attention to the computational aspects of the project. We are building models using several signals. One of our goals is to maximize prediction performance, and to do this one generally uses complex models, which demand high computational power, and give it as many features as possible for input. To balance this, we tried to refrain from adding non-relevant signals, and design models of high enough capacity, but not more. This will have cascading effects once these models are placed in production because the less signals we have, the less processing we need to perform to get the input feature vector for each user. The same goes for model complexity: lower (while meeting our performance requirements) is better. Another aspect of sustainability to consider is specific to the task of finding the best model, i.e. hyper-parameter search and repetitive execution. To address these, we went for heuristics driven search thus limiting our search space as well as number of executions needed to find a suitable model, which fit well for our case.

In addition to this, building a better product requires a focus on the initiatives that will bring the most positive value for both producers and consumers. This focus is defined by the allocation or people's time and effort, as well as financial and economic investment. Reaching this requires an informed reflection on internal practices and product usage. Part of that is understanding which specific features provide the highest value, while being sustainable for the company to deliver in the present, and in the future. The results of this research can aid in that aspect by helping uncover those initiatives and shape an approach to measure and reflect on them.

1.6 Contributions

We outline the following contributions:

- Analysis of user activity, demographic, financial, and performance data for user understanding in a music streaming service.
- Analysis of churn using the same data in a music streaming service.
- Comparison of machine learning models for churn prediction in a music streaming service.
- Proposition and evaluation of an alternate technique to enhance ensembles of Decision Trees (DTs).

1.7 Delimitations

As with any research study, this project is constrained by time, and other resources. As such, we define the following constraints in the scope of work:

- Establishment of causal relationships: while the aim of the study is to understand if and how certain features might affect user churn, this understanding will be limited to studying correlations, interpreting feature importance in models, and mining patterns from data. No outcome is conclusive, but they do serve as basis for future validation tests, e.g. A/B testing, bandit algorithms.
- Heuristics guided hyper-parameter search: the performance of models with hyper-parameters can vary widely depending on said parameters. For this study, both grid search, and heuristic (manual) guided search are used to limit search space and time.
- Models tested: due to time constraints, only four models will be tested, and evaluated for the task of churn prediction. Therefore, there may be other models that could perform better, but they were not evaluated.

1.8 Outline

We have already addressed the problem, main goals, and hypotheses for the study in this chapter. Following, we describe how the rest of the document is organized. In chapter 2, we address churn analysis, discussing the relevant theory and scope of work, as well as the models and methods used. Chapter 4 is about the methodology of the study, from data pre-processing to coming up with the final feature set used for modeling and rule mining. Chapter 4 covers the design of experiments to answer our hypotheses questions. In Chapter 5, we present the results of the experiments, which we discuss in Chapter 6 along with possible future work.

Chapter 2

Background

2.1 User Engagement

Both retention and activation are often associated with continuous engagement of users with a service. Churn, after all, is defined for us as lack of engagement. In order to understand churn, we first try to understand how users interact with a service and how those interactions can be measured. The amount of research on this topic, along with its impact on improving retention and reducing churn, is wide and varied [3, 4, 14, 15, 16, 17, 18, 19, 20], both in the social and computational sciences. Several researchers have tried to understand specific aspects of user engagement, while very little work has gone into defining user engagement itself. This is not surprising, considering the broad set of contexts in which one can use the term.

For instance, when looking at social media, Wasko and Di Gangi [15] tried to define engagement as two distinct types of experiences: one derived from social engagement and the other from technical features. The authors found that while frequency can be an important factor in measuring user engagement, the quality of the experiences tends to be more relevant.

OBrien and Toms [16] conducted research in an attempt to define the term engagement. From their work, engagement was proposed to mean a “quality of user experience characterized by attributes of challenge, positive affect, [sic] endurability, aesthetic and sensory appeal, attention, feedback, variety/novelty, interactivity, and perceived user control" (adapted from [16]). While this definition is very much user-experience design centric, there are some relevant aspects that it highlights, such as positive affect, and feedback. After all, most products seek to leave the user with a positive affect, either because they managed to accomplish a given task, or because they managed to have the emotional experience they sought. Interestingly, the authors contrasted user engagement from flow, which has often been considered a super-set of user engagement by some. They explain that unlike flow, engagement can occur without long-term focus and loss of awareness of the world outside. To give an example, searching for answers in a Question and Answers (Q&A) website could take as long as one minute. During such a short period of time, the user would have not experienced a flow, but they would still have had an engagement experience with the site, and from that experience they would have derived a positive or negative affect. Therefore, duration of a session with a product alone would not necessarily be a good measure of engagement. As another outcome of their research, the authors established from their experiments that engagement sessions can have multiple engagement episodes, and that the return of users was a high predictor of system success. Quality is another aspect that can define engagement as well as influence retention. From empirical assessments done by Tractinsky [3], it is known that aesthetics and usability are not only important for quality assessment, but are also culturally variable.

But what is, then, user engagement? It is hard to define, but from the reviewed body work, one could suggest as synthesis that user engagement would be the measurement of a user's reaction to a stimulus provided by the service, with the intent of understanding the effectiveness of said stimulus, and/or the user's preferences.

Lehmann et al. [4] tried to provide formal and comprehensive models for understanding user engagement. In their definition, they considered user types, and temporal aspects. First, and foremost, the authors establish a relationship between user engagement and aspects of being captivated and motivated to use a product or service, which aligns well with our synthesized definition. They also establish that specific measures for any given application or service would be highly dependent on the domain. Then, they proceed to put forward a categorization of three methods to measure user engagement: (a) self-reported engagement (e.g. questionnaires), (b) cognitive metrics (e.g. heart rate monitoring when performing a task), and (c) online behaviour metrics (e.g. usage time, return frequency, click-through rate). Their study focused on the third, as it is the most feasible and practical for digital services, and while it fails to provide reasons for the measures, it lacks the drawbacks of subjectivity and cost which the first two methods have, and often provides good proxies for actual user engagement.

Lehmann et al. [4] then proposed three main types of metrics, for measuring engagement: popularity, activity, and loyalty. In their study, they found no correlation between the three, although when clustering different services, some patterns of combinations of the three emerged. For instance, as stated earlier, the time each user spends on a Q&A site each time they visit it is low, compared to a movie streaming site. For these two, loyalty, and popularity would be measured in different ways, and one might find one metric to be more relevant than the other.

From their analysis, three sets of models were proposed, to explain the data:

1. General: based on specific engagement metrics, e.g. popularity, loyalty;
2. User-based: accounting for user groups, e.g. tourists vs VIP;
3. Time-based: accounting for time aspects, e.g. weekday vs weekend use.

Conceptually, the models prescribed by Lehmann et al. [4] characterize different aspects of user engagement. From them, we learn that there are various ways to look at engagement, and each service should find the lens of view that suits them well. We use this as a basis for feature engineering for our tasks of predicting and understanding churn. For our metrics, we try to elaborate on those measures that best reflect how users respond to different features, product offerings, and the quality of the service.

2.2 Related Work

We have two main tasks to conduct: churn prediction and Quantitative Association Rule Mining (QARM). In this section, we briefly discuss some of the reviewed literature on each of these subject matters, with emphasis on the different ways they have and can be carried out, noting the most recent trends.

2.2.1 Churn Prediction

Churn prediction has been attempted in various studies, with different aims, ranging from model comparison to temporal signal usage, and time horizon prediction [21, 22, 23, 24, 25, 26, 27]. For instance, there are authors that have focused on comparing different techniques and approaches in a particular domain [21, 22, 26]. Vafeiadis et al. [21], for one, conducted a comparison study between machine learning techniques for customer churn prediction in the telecommunications domain. In their study, they compared single models to boosted ones. Overall, Artificial Neural Networks (ANNs) performed best for their data set in the single model category, while Support Vector Machine (SVM) had the best result in the boosted category. In both categories though, RF had very high performance. The authors also found that all boosted models had significant improvement in the F1-score metric relative to their single version, with 11% increase observed in SVM, and only a slight improvement in accuracy. Their tests were based on a public churn data set, where most features were of numerical, aggregate nature, e.g. total number of calls and total call time in minutes. Some of them represented temporal aspects, e.g. how long has the customer been active, or total number of calls made at night. In a similar study, Keramati and Marandi [22] compared meta-heuristic methods, machine learning, and data mining techniques. Among their tested methods, they highlighted ANN, K Nearest Neighbour (KNN), DT, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO). To use meta-heuristic models, i.e. GA and PSO, the chromosome and particles were designed to be comprised of the required model parameters, and the fitness function evaluated the performance of the tuned parameters on the data. Similar to Vafeiadis et al. [21], their ANN model performed best in all instances, with the exception of two. Their data included usage features, such as calling time, and service quality features, like the number of complaints.

Other authors have studied different formulations to the problem, centering on temporal aspects [23, 26, 27]. Rothenbuehler et al. [23] studied the application of Hidden Markov Model (HMM) to churn prediction. They tested and evaluated user activity, using general data, such as number of sessions per day. This was done for a mobile game that used a freemium model, so churn was defined as loss of interest for a period of 14 days, after which most users did not ordinarily return to the service. For proactive action, a customer was considered a churner if they left the platform six days after prediction. As such, their data did not span the entire life span of the customers' existence, but rather their activity over the past 24 hours, and for active customers only. Their HMM performed just as well as their ANN and SVM models in most cases. Since the HMM model was lighter in space and computation compared to other two, and it had the benefit of providing a description that could be easily interpreted by people, e.g. state transition information, it was considered a valid alternative. Further tapping into the temporal aspect of data, the authors utilized moving averages to define features, as opposed to actual values per day, as these may suggest a more robust trend in user behaviour. Ali and Arıtürk [27], on the other hand, built a framework for generating training data that gives churn prediction ranking for each user at different time horizons, instead of a single one. Rather than employing Survival Analysis (SA), which is a statistical method commonly used for this task, the authors opted for a Logistic Regression (LR) model, and pre-processed their data to enhance information. The framework was claimed to have had (1) improved prediction accuracy of both their LR and DT models, enabling identification of impact of environmental factors and (2) provided insights about churn drivers. Their method consisted of using independent classifiers for each future time window. The authors included recency, frequency features in their data, and value of transactions, as commonly done in banking.

From these studies we identify distinct approaches being tailored to a specific problem formulation, with the choices of methods based on the data and target outcome (e.g. time to churn). With this philosophy, with conceptualized our own problem formulation.

2.2.2 Quantitative Association Rule Mining

Association rules were initially used for binary variables, e.g. user bought or did not buy an item. Quantitative Association Rules (ARs) require a different approach. If one uses numerical variables, then the number of possible values can quickly expand; using ranges can help reduce the number of possible cases, however, it can still be expensive. Adhikary and Roy [28] presented a synthesis of the trends in QARM, which we summarize.

For starters, they categorize all existing approaches into five categories: partitioning, clustering, statistical, fuzzy, and evolutionary. The partitioning approach consists of converting quantitative attributes into boolean attributes. An example of the approach taken by Srikant and Agrawal [29], which consisted of splitting each quantitative attribute into disjoint partitions, and then mapping each partition to a boolean value. This approach has a trade-off in that too much partitioning generates more variables, and reduces support for each range; too few means information loss (see Fukuda et al. [30] and Li, Shen, and Topor [31] for similar techniques). Clustering approaches use FP-trees, DGFP-trees, and similar data structures to divide data in an N-dimensional space into cells. This solves the low frequency issue of partitioning approaches. Some of these techniques find clusters, and then use them as item sets to find relationships. Clusters are formed, and then each cluster is mapped into (possibly) overlapping intervals of the data, which was asserted to be a good resolution to the minimum support and minimum confidence issues in partitioning approaches [32]. The drawback is that only positive rules can be generated from this kind of approach.

Statistical approaches, in turn, rely on statistical measures like mean and standard deviation. Kang et al. [33], for instance, convert quantitative features into binary, perform binary association rule mining, and then convert it back to quantitative association rules. They made use of standard deviation, by creating partitions with the smallest possible standard deviation, i.e. high value cohesion. This requires making multiple passes on data, which can be computationally costly for large data sets.

Fuzzy methods, for one, make use of the Apriori algorithm for Association Rule Mining (ARM). Generally, they can only find positive rules, with usually two antecedents and one consequent. Evolutionary methods rely on single and multi-objective GA or PSO. In these methods, rules form the individuals in the populations, and they can be comprised of single or multiple quantitative (and categorical) features in both the antecedent and consequent. Their main benefit is that they do not required any data preparation. They are capable of finding positive and negative rules, without discretizing attributes, and hence, have become more popular in recent times.

It is worth noting, as mentioned by Gosain and Bhugra [34], that ARs do not imply causality, nor do they imply correlation, i.e. X $\rightarrow$ Y does not mean Y $\rightarrow$ X, as it happens in correlations.

2.3 Models

In this section, we describe relevant machine learning, data mining, and statistical methods for churn prediction and association rule mining, and user understanding. We conclude with a discussion of advantages and disadvantages of each, the choice of model we made, as well as our reasoning for said choice.

2.3.1 Logistic Regression

Logistic regression is a prediction model used for classification. It works similarly to linear regression, in that it tries to find a set of coefficients to pair with each input feature; however, instead of giving a numerical output, it gives a probability of membership of an input vector $x^{(i)}$, to a class in a dichotomous (binary) or nominal (more than two) variable $y^{(i)}$. The activation function σ is called the *logistic sigmoid* function. It depends on the weight vector θ, and in the input feature vector $x^{(i)}$, $\sigma(\theta^T x)$. See Equation (2.1) on the next page

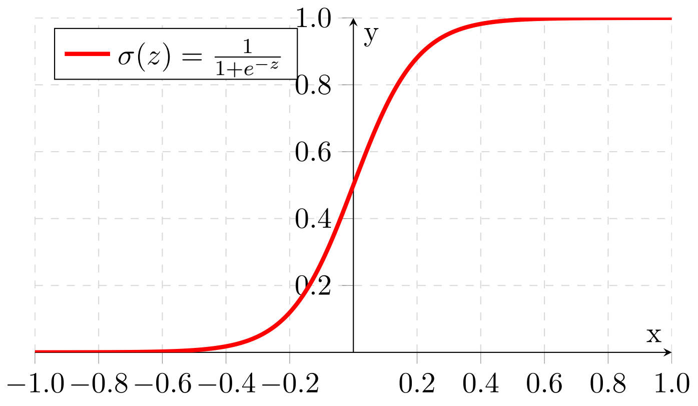


Figure 2.1: Logistic sigmoid function: used in logistic regression to compute the probability of membership of an instance/example into a class, where $z = \theta^T x$

$$\begin{aligned} P(y=1|x) = h_\theta(x) = \frac{1}{1+exp^{(-\theta^T x)}} \equiv \sigma(\theta^T x), \\ P(y=0|x) = 1 - P(y=1|x) = 1 - h_\theta(x) \end{aligned} \tag{2.1}$$

The *logistic sigmoid* function squashes the value of $\theta^T x$ to the range [0, 1]. For binary classification, the optimization problem is to find the set for values for θ for which if the given input $x^{(i)}$ belongs to the binary class 1, then $\sigma(\theta^T x)$ will be close to 1; and if it belongs to the binary class 0, then $\sigma(\theta^T x)$ will be closer to 0. It's an S shaped function, whose optimization is measured by the cost function $J(\theta)$. See Figure 2.1, and Equation (2.2).

$$J(\theta) = -\sum_i (y_{(i)} log(h_\theta(x^{(i)})) + (1-y^{(i)}) log(1-h_\theta(x^{(i)}))) \tag{2.2}$$

To find the best fit θ, we minimize our cost function $J(\theta)$. To do that, we can use the gradient of the cost function $J(\theta)$, with respect to θ. See Equation (2.3).

$$\frac{\partial J(\theta)}{\partial \theta_j} = \sum_i x_j^{(i)} (h_\theta(x^{(i)}) - y^{(i)}) \tag{2.3}$$

The way the function determines membership to the binary class 1 is by checking if $\sigma(x^T\theta) > 0.5$. The algorithm for minimizing the cost function, $J(\theta)$, is relatively simple. The coefficient weights assigned to each feature variable can indicate the relationship between said feature and the outcome variable, provided the confidence interval values are within reasonable range [35]. A more detailed reference on Logistic Regression can be found in Rodriguez [35], Peng, Lee, and Ingersoll [36], and Cramer [37].

2.3.2 Artificial Neural Networks

Despite having become more widely used over the past decade or so, ANNs have actually been around since the 1950s [38]. Frank Rosenblatt first came up with the concept of the learning perceptron. While it showed promise in its ability to solve certain basic problems, using addition and subtraction, popularity of the perceptron waned about decade later when Marvin Minsky and Seymour Papert published a thesis on the limitations of the perceptron [39]. The misconception at the time was that the limitations described by Misky and Papert, which were the perceptron's (1) inability to solve an exclusive-or circuit and (2) their high computational cost, meant that ANNs were not a viable method. At a later stage, the back propagation algorithm came to prominence. This algorithm made is significantly easier, and faster to train ANNs to learn. Then, using backpropagation and a multi-layer perceptron, a solution to the exclusive-or problem was found, sparking interest, once more, in ANNs. With the recent growth in computational power over the past two decades, ANNs have increasingly been growing in adoption for their tremendous ability to solve classification, pattern recognition, and sequence problems. Yet, even with these advances, ANN are still relatively more complex to train [40, 41].

ANNs come in several flavours, and styles. Two of their main features are the neuron, and their architecture. The logistic sigmoid function described in Equation (2.1) on the preceding page is one of several types of neurons that exist. Sigmoid neurons can suffer from a vanishing gradient problem. Alternatives such as Rectified Linear Unit (ReLU) [42] and Exponential Linear Unit (ELU) [43] do overcome that issue. ELUs, for instance, and make it easier to train by introducing identity for positive values and negative values which allows them to push mean unit activations closer to zero with lower computational complexity than alternatives like batch normalization; however, they can suffer from other problems, like the exploding gradient. These issues are described in detail by Pascanu, Mikolov, and Bengio [44]. All in all, it is not always clear which activation function or type of neuron will yield best results for a given problem. Generally, certain pre-processing steps should be taken to train a neural network, like mininum-maximum normalization.

Neural network architectures are extensively discussed by Demuth et al. [45]. The most common types of architectures are Feed Forward Neural Network (FFNN), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN). The first, Feed Forward Neural Network (FFNN), are the most common, and basic type of network. They are comprised of at least one multi-preceptron layer, fully connected to the input, and an ouput layer. The second, Convolutional Neural Network (CNN), have grown popular in use in the image processing domain. Convoluational networks are well suited for 2-dimensional grid data, which is why they work well in image recognition and transformation. The third kind of neural networks are Recurrent Neural Network (RNN). These were especially designed to tackle problems of a sequential nature, e.g. speech translation. RNNs are very flexible in their usage. They can be used to predict the next item in a sequence, e.g. the next word in a sentence, or to map a whole sequence to another. Conversely, they can be harder to train than FFNNs.

2.3.3 Decision Trees

Decision trees are considered part of a family of tree based methods. They can be used for both prediction tasks of regression and classification. Since our proposed alternate technique applies to Random Forest (RF), which are built on decision trees, here we discuss the fundamental concepts of tree based methods and their ensembles. For a more thorough treatment on this topic, we refer the reader to Breiman et al. [46] and James et al. [47].

Unlike mathematical models, such as logistic regression, decision trees form sequences of rules to either estimate, or classify an input instance, $x^{(i)}$. The rules in a tree are called branches. Each branch in a tree starts from the root, and leads to a terminal node, i.e. leaf. In regression problems, this node determines the value to be assigned to input instance, and for classification problems, the class the instance belongs to. Here, we focus on decision trees used for classification, though they work analogous to the ones used for regression.

A decision tree sequentially divides the training data into linearly separable, and non-overlapping regions. Each instance in the training data belongs to one, and only one region. The regions are found through a greedy algorithm called recursive binary splitting. At each node, starting from the root, the algorithm picks the feature that would provide the best split. The best split is one that will give the highest information gain. In a classification task, this could mean reducing the classification error.

To decide on a split, the algorithm checks for each feature the splitting point that would reduce the total classification error. However, at each node, only the instances of the training example that fall under that node will be considered for the split. Once the feature and split value are selected, all instances whose value for selected feature are less than or equal to the split point will be assigned to the left branch of the tree. And all instances whose value for the selected feature is greater than the split point will be assigned to the right branch of the tree. In a terminal node, the class of the nodes that fall under its branch is simply that of the majority of the training instances that fall under the node (in regression trees, the predicted value would be the average of the values in the samples that fall under the node).

As stated earlier, the measure by which features are selected for splitting a node could be the classification error, i.e. accuracy. In practice though, because accuracy is not as sensitive to branching and depth of a tree, *gini index* and *cross-entropy* are used instead. Gini index is a measure of purity of a decision tree. It's formula is illustrated in Equation (2.4) on the facing page, where m is a region (i.e. node), and $\hat{p}_{mk}$ is the proportion of samples in each leaf that belong to a class other than that of the majority of the instances in the leaf. Intuitively, the smaller the gini factor, the more certain the leaf's classification will be; i.e. small gini indicates high classification confidence, and thus lower errors. Cross-entropy is a measure very similar to the *gini index*. Its formula is in Equation (2.5) on the next page.

Instead of multiplying the proportion of instances that belong to a class other than that of the majority of instances in a leaf node by the proportion of classes that do, the *cross-entropy* multiplies it by its log. Like the *gini index*, lower values indicate higher adhesion of the sample instances to a single class, and thus higher classification confidence.

There are no mathematical, nor experimental proofs that one measure is better than the other, and both are equally used in practice.

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{2.4}$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log(\hat{p}_{mk}) \tag{2.5}$$

Figure 2.2 on the following page is an example of a decision tree. At each node, we can see the feature used for the split, along with the splitting value. For example, the root node uses the feature *Sepal Length*. This means that this feature gives the highest information grain when used as a first step to split the data set into two regions. The *gini index* is also given. Aside from this, the node has information about the number of samples used for the split, and the number of samples that belong to each class. The root node uses all samples in the data set for the split

The following are some of the advantages commonly attributed with decision trees:

1. Ease of interpretation.

2. Higher modeling capacity compared to linear methods.

3. Nice graphical presentation.

The first point can sometimes be a requirement in certain domains where a clear explanation the the workings of the algorithm are a requirement, like medical diagnosis, or industrial chemistry research.

The problems attributed to decision trees are that, because they're built on a greedy algorithm, they may either overfit, or miss good splits downstream in a branch and underfit. To address overfitting, decision trees can be pruned. Pruning is a technique to cut off nodes and branches of a tree that do not add significant improvement to overall performance. A good treatment of the the existing pruning strategies can be found in Mingers [48]. Alternatively, and in practice, ensembles are used to address the shortcomings of decision trees.

Ensembles are a machine learning technique that consists of combining several weak learners to form a better one. We briefly discuss three methods to using ensembles with decision trees. A fair treatment of these can be found in James et al. [47].

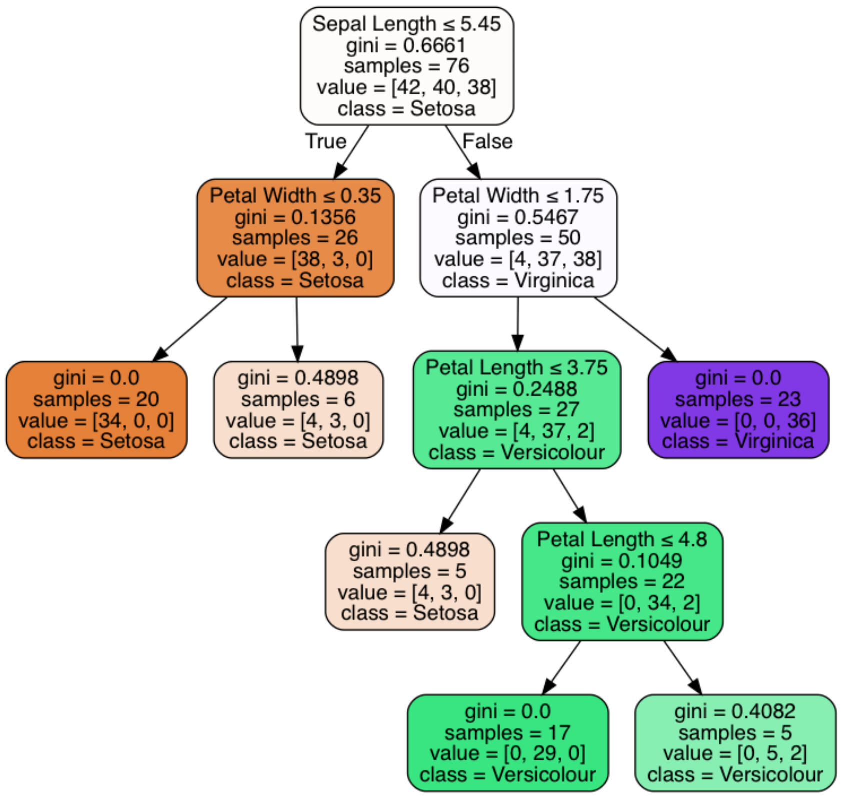


Figure 2.2: Example of a decision tree, formed with the public Iris data set. Samples indicate the number of instances that fall under the node, gini the error. In the leaf nodes, value list indicates the number of samples that fall under the rule, and the class is their classification.

Bagging

Bagging, also referred to as boostrap aggregating, is a method of building an ensemble of weak learners by using different sample sets for each one. For decision trees, this would mean that each tree in an ensemble may end up with a different root node, and branches.

Since trees are prone to overfitting, pruning is not applied when using bagging. The resulting ensemble is made up of multiple trees, each with high variance, and low bias. To perform classification, each decision tree votes on the class they predict an example instance belongs to. Finally, to classify the example, a simple technique like majority vote is used. As the classifiers are different, and there are several of them, this combination reduces the variance of the model, making it more stable.

However, since more than one tree is used, and trees can have differing structures, some of the interpretability that was inherit to a decision tree gets lost. Nonetheless, random forests can still give a measure of understanding, called feature importance. Feature importance is based on the information gain, i.e. total reduction in classification error, that is obtained from each feature, across all trees in an ensemble.

Random Forest

Bagging tries to improve decision trees by introducing a randomness to data selection, and using multiple weak learners. Random forests take this concept a bit further.

Instead of using the feature that gives the highest information again for a node split, a random forest algorithm picks a subset of the features, and randomly chooses one of them. This subset can be $\sqrt{p}$, $\ln p$, where p is the number of features, or some other subset.

The aim of this is to reduce the correlation between trees in an ensemble, since the greedy algorithm will tend to select the same sequence of features to build branches in trees, even if different samples of data are used at a time. This also gives a better chance to the other features of being selected, and potentially overcoming the short-sightedness of the greedy approach. On average, $\frac{p-m}{p}$ of the splits will not even consider the strongest feature, where m is size of the feature subset.

Boosting

Despite being an improvement over bagging, random forests still suffer from one drawback: each weak learner is built independently of the others. Boosting, on the other hand, is a sequential learning method. Sequential learning methods tend to perform better than their counterparts.

Instead of fitting N independent trees, boosting will build trees one by one, each one designed to improve the previously built set of trees. Unlike bagging, and random forest, boosted decision trees are not built to minimize the classification error in each tree, but to further reduce the error of the ensemble.

Since the algorithm builds each tree with working knowledge of the existing ones, much smaller trees, i.e. stumps, can be used, and have been found to work better in practice. Intuitively, as each sub-tree is added to the previous set, we end up with an additive model.

A drawback of boosting, particularly those built on stumps or small trees, is that the rules formulated are harder to interpret.

2.3.4 Neural Augmented Trees

While they may perform well for many problems, the criteria for class selection in a classification RF is majority vote. A key advantage of this simplistic approach lies in its efficiency. Each tree gets a vote, and all the algorithm has to do is count the votes towards each class. Whichever class had the highest vote count is chosen as the predicted class. In this thesis, we propose an alternative to this.

Conceptually, having an ensemble of weak learners of high variance leads to an agglomerate learner with lower variance. Hence, majority vote is a practical and theoretically valid mechanism for ensembles of decision trees. However, when it comes to problems that involve the usage of multiple and often different models, researchers tend to take an alternate approach: instead of using majority vote, an algorithm is trained to learn how to weight the output from each model in the ensemble, thus correcting for generalization error in each model. This meta-learning approach, while being more sophisticated than simply taking majority vote, introduces a new learning task to the problem which can be optimized with known techniques. It was first proposed by Wolpert [49], who coined it the term *stacked generalization*, and proved that is a theoretically valid way to improve the performance a single, and multiple generalizers, and to be a better alternative to using simple approaches like majority vote and averaging.

Originally in stacked generalization, or stacking, each generalizer in an ensemble was trained and tested independently on a given partition of the data, and taken as if they're the best version of their model type. This can be done using cross-validation training, where each sample partition is mapped from the original feature space to a second one using a model for which it was an out-of-core partition. Later, the concept was developed by Ting and Witten [50] and Džeroski and Ženko [51]. These authors demonstrated that these partitions could either be disjoint, or formed through a method like boosting. Ting and Witten [50], the authors even proved that using stacked generalization with bagging should always yield higher predictive accuracy than simply using bagging. Bagging and boosting are widely used methods for combining methods of the same type. Stacked generalization, on the other hand, is used to combine different models, and it can introduce non-linearity by using a non-linear model as the meta-learner.

RFs use bagging, which means that each DT is built using a random subset of the data that was sampled with replacement, and then apply majority vote. Though the models are not of a different nature, it would still stand to reason that if one were to train their output in a meta-learner, using the output of the original problem as a target, we could reduce the generalization error of the RF model. After all, each DT is built using only a sub sample of the data. Therefore, if we take the probability guess of each tree, we include the guesses for the trees that were trained without a given sample, and with it as well.

What we will attempt to do is to improve the voting mechanism in RFs, by training a meta-learner on the output for each DT. Specifically, we will attempt to learn the best way to use confidence votes from the trees in our ensemble to approximate the actual output from our data. Essentially, we wrap our original features into a new dimension. Like in stacked generalization, this dimension has two interesting properties:

- The value range of each input feature lies naturally in the interval $[0, 1]$, which has shown to be practical for minimization algorithms like Gradient Descent.

- Each feature, representing a tree, is expected to vote consistently given a range of values in the original input space.

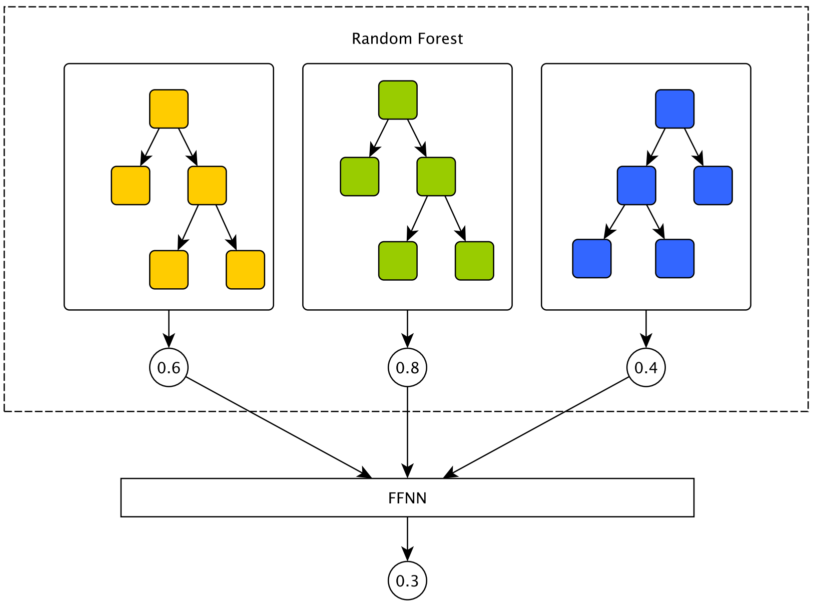


Figure 2.3: Neural Augmented Trees: A Random Forest model with three Decision Trees connected to a Feed Forward Neural Network

And so, to improve performance for those cases where the right answers would come from the minority in the ensemble, which could be a single DT, we shall take the probability values from each tree, and feed them to an ANN, and see if it can learn to overcome the shortcoming of majority vote as an ensemble quorum method, as is done in stacked generalization. Because we intend to use an FFNN as a meta-learner, we call this technique Neural Augmented Trees (NAT). Neural Augmented Trees (NAT) hyper-parameters depend on the hyper-parameters of the meta-learner that's used, which in our case is a FFNN. Figure 2.3 depicts how this method would work.

Where this technique differs from originally proposed cross-validation based *stacked generalization* is that our prediction set for each sample is not only from the models for which that sample was an out-of-core sample, but from those where it served as a training sample as well. However, as stated earlier, Ting and Witten [50] demonstrated the technique should work for models built using bagging, as is our case.

We will test this technique through empirical evaluation, by measuring performance difference between base RF model and its NAT equivalent.

2.3.5 Meta-heuristic Methods

Meta-heuristic algorithms can be defined as algorithms that guide a heuristic, i.e trial and error process, to solve a problem [52]. They are widely used in optimization problems for engineering, biology, chemistry, and other application fields. However, they can also be applied to problems like churn, where traditional machine learning methods are used. For example, Keramati and Marandi [22] applied GA and PSO algorithms to predict churn. Since neither algorithm can represent a function for prediction, the authors formulated their optimization problem as one of finding pairs of coefficients and their power, for each of the features in the data set. They compared their solution to ANN; the latter outperformed both in most cases.

An area where meta-heuristic methods have been successful is rule mining. Early in 2003, some researchers attempted to use an evolutionary computing approach for rule ranking, with good results [53]. The rules were extracted from the data using probabilistic induction, and a genetic algorithm was used to select the best among them for the task of classification. In 2007, Salleb-Aouissi, Vrain, and Nortet [54] introduced a genetic algorithm approach that did not require discretization for association rule mining on quantitative data, along with an open source implementation called named *Quantminer*.

As a first step, *Quantminer* computes rule templates, consisting of a number of numerical feature ranges and values for categorical variables in the antecedent and consequent, using the Apriori algorithm. For the numerical variables, they are first instantiated with a range that is delimited by the minimum and maximum values; for categorical features, each value is instantiated into a new variable. Then, for each template, it finds the best interval of values (or value) that fulfill a minimum support and confidence criteria. Through a GA cycle, it creates new individuals choosing random values as range delimiters that cover a width that's large enough to have the required support. During cross-over a new individual can inherit bounds from one of the parents or mixin bounds from both parents becoming wider or smaller in value range. During mutation, the algorithm can alter the upper or lower bound at random, while discarding no more than 10% of examples covered by the rule. The main metric of fitness is defined as gain, and it establishes the goodness of a rule with respect to its support and confidence, so that rules with higher support and confidence are ranked higher, and those that don't meet the minimum confidence are heavily penalized.

$$Gain(A \rightarrow B) = \Theta(AB) - \Phi * \Theta(A) \tag{2.6}$$

Equation (2.6) is the formula for gain, where Θ is the support, and Φ is the minimum confidence. Following this notion, similar approaches came later, based on GA [55] and PSO [56, 57]. The algorithm introduced by Alatas and Akin [56], for instance, did the rule mining automatically without the need for minimum confidence and support thresholds, while the other proposed by Beiranvand, Mobasher-Kashani, and Bakar [57] supports more than one objective, and it tries to balance confidence, comprehensibility and interestingness of rules. For an introduction of GA and PSO, the reader is referred to Deb et al. [58], and Shi et al. [59], respectively.

2.4 Model Selection and Use

Several approaches have been tried and tested for the task for churn prediction. Some using more explanatory methods than others. Logistic regression had been traditionally popular for this. However, interpreting its coefficients is limited, as is the structure of the problem, for it uses a single linear function to map the feature space to the outcome variable. This is not a problem in cases where a linear decision boundary does in fact exist in the data. Despite this, we tested logistic regression, to a limited extend, without performing feature transformation, e.g. polynomial transform, to introduce non-linearity. Tree based methods tend to be used for three main reasons: (1) they provide a simple explanation of the data, (2) have higher modeling capacity compared to linear methods, and (3) they are relatively easier to train than other methods. Given the performance enhancement of ensembles over single methods, random forests will be selected as a method of evaluation in this work. Despite not giving a simple clear explanation, as a single decision tree, we can still extract the most relevant predictors as measured by feature importance. We believe this to be sufficient for our problem, considering the volume of users at hand. While boosted trees can theoretically provide superior performance, as a method, due to time constraints they were not tested. It should be noted that decision trees rely on the notion that a problem can be solved through recursive partitioning of the feature space, one predictor at a time. If this proves false in the data set, they are expected to perform poorly. Now, as we intend to see how a more powerful, and less interpretable method would perform, ANNs, PSO, GA, and similar approaches would be suitable candidates. However, in prior studies, the meta-heuristic approaches have had little gain over ANNs as prediction models, while lacking a clear problem formulation structure, thus needing extra work in design, and training. For one, both PSO and GA require us to make assumptions about the model that describes the distribution. While it may very well be polynomial, we saw no benefit in investigating this, due to time constraints. As such, neither of them were tested for the task of churn prediction. And as for Hidden Markov Model (HMM), our problem formulation is not defined by time steps or stages, and so we saw no point in testing the method for our case.

Rothenbuehler et al. [23] modeled churn as a moving target, with recency embedded in the features, while Ali and Arıtürk [27] also tried to predict time to churn at different observation and churn windows. Despite taking different approaches to the same problem, these studies have a few factors in common: (1) the use of activity data combined with quality of the service in certain way; (2) the inclusion of temporal dimension to data, either in the features, and/or the dependent variable. This work aligns with the formulation of user profiles by Lehmann et al. [4]. We use this as an inspiration for deriving features to describe users on the data set. We are left then with three standard approaches to use on the task of churn prediction: LR, RF and ANN; and NAT as an alternate one to be tested.

A field where meta-heuristic methods have been particularly useful is ARM, and both Gosain and Bhugra [34] and Adhikary and Roy [28] describe this trend.

As they have been successful at this task, we shall employ them to our problem to derive rules that describe user behaviour in relation to retention. In particular, we shall make use of algorithm proposed by Salleb-Aouissi, Vrain, and Nortet [54]. Despite there being more recent and evolved algorithms, time constraints would not allow us to implement and verify them before using them on our data set.

Chapter 3

Methodology

In order to study users' activity to understand their behaviour, researchers and practitioners make use of different combinations of analytical methods, e.g. correlation analysis and clustering. These methods are applied to theorize hypotheses about users' behaviour, and factors with which researchers can create models of users, i.e. profiles. The hypotheses, for one, can be validated or refuted by running experiments, e.g. A/B testing and bandit algorithms [60], or by using offline methods. The models, on the other hand, are used to predict which users are likely to remain, and which are more likely to churn. These models can be general, centered on the user's historical activity, or even temporal, i.e. trend oriented [4].

Churn, as a term, can have various definitions. A specific formulation of churn is predicting whether users will remain with a service after a certain period of time. For example, given that a user registers for a music streaming service, one might try to predict based on their activity if they will stick with the service after a month. This is the definition we use, but with a shorter window of one week. A significant amount of work has been done in churn analysis with different problem formulations and tried techniques [21, 22, 23, 24, 25, 26, 27]; much of it focused on the telecommunications industry, or in banking and financial services. Our survey of this work guided our reflection on the different steps of our problem design, with particular attention to feature engineering and outcome definition.

In churn prediction, historically both LR and DTs have been popular methods of choice, because the first is simple and easy to train, and the second provides a set of decision paths that can help a business understand what factors drive users' behaviour [25]. These can be activity related, such as the features of the service the user explored, and how much time they spent on each; financial, in that users may subscribe or be registered to different product tiers, and transition between them; temporal, in that users may exhibit repeated behaviour on a time scale; or performance related, such as playback latency, or transaction delay. The drawback of LR is that it has limited capacity, by nature of being linear; for DTs, it is mainly their sensitivity to sampling techniques and value ranges. Both of these problems are commonly addressed by using an ensemble of DTs, i.e. RF. RFs are relatively simple models, with very powerful capacity. Yet, as an ensemble, they use a simple quorum technique: majority vote. Thus, we propose as an alternative using a learning model, instead of majority vote, to improve model performance. ANNs, in turn, have been growing popular for their high predictive performance. Alas, they have the drawback of lacking easily interpretable parameters, which makes them unpopular with tasks where the outcome as well as an explanation for it are equally important. Also, they often require substantially more computational resources to find the right architecture and training parameters [61, 44].

In ARM, algorithms like Apriori are used to find relationships between items in a database (see Borgelt and Kruse [62] for an introduction to the algorithm). For that, they are very popular in the task of basket analysis. However, when we have continuous variables instead of categorical items, those algorithms fall short, and we need alternative methods to deal with the continuous variables. Such methods are called QARM methods [63, 54] and, since our feature space is mostly comprised of numerical variables, we shall use them for our problem. QARM techniques do to numerical features what standard ARM does for items in baskets, in a context where items represent continuous features with a given range of values or a specific value of a categorical feature. In a sense, this is conceptually similar to coming up with fuzzy rules describing a population. The most recently successful QARM techniques rely on evolutionary algorithms, for practical reasons like their flexibility in exploring the search space and scalability in handling large data sets [54, 56, 57, 55].

Data for this study was gathered from logs, which are produced by clients via events triggered by users interacting with the service's applications. The methodology approach of choice is the quantitative method, as it stands on objective measures of performance that can be used to compare and contrast solutions, which we intend to do with our prediction models, and is widely used in the field of classification modeling.

As a method of evaluation, for prediction we use the empirical method, assessing results from prediction models on the basis of their performance on unseen data. We use cross-validation to ensure stability of the results, as opposed to train-validate-test data split. We did this to avoid having less data for training. On the rules mined, we use inductive reasoning to appraise them in relation to other gathered evidence, such as correlations and feature importance.

3.1 Study

As a first step in the study, we began by analyzing and reviewing earlier work done on the population, to find commonly occurring patterns. Through this, we found hints of diverging behaviour between population groups, and relevant indicators of long term churn. Since we wanted to focus on short term retention, we replicated some studies and analyses, previously done on the population, to see if the indicators would be the same.

Part of the data understanding process involved source code reviews, through which we had to verify the meaning and significance of certain log fields. The source code we refer to is that of client applications and internal data processing pipelines. Following this, we conducted statistical analyses on the features we aimed to observe. Our objective was to understand their reliability and use, since not all platforms report information in the same manner, due to the design of the interaction methods rendered to each, the fact that several of the measures in testing were somewhat recent, and still evolving in their definitions.

Because the nature of the signals under analysis differed, they were stored in disparate lakes, spread across different data sets in a SQL-like data warehouse. We joined data using anonymized user IDs.

Parallel to this process, we conducted extensive literature review of churn analysis and the most common machine learning methods reported to have successful applications with the problem. In doing this search, were interested in two particular things: (1) selecting a technique to extract insights from data and (2) techniques for doing prediction.

For mining insights, we came to ARM for categorical and numerical features. Our review of the state of the art showed meta-heuristic methods to be go to choice for them. We arrived at two fairly similar techniques for QARM, one based on GA and the other on PSO. Despite the latter being more recent, the former contained an open source implementation, which could be immediately used, and so we opted for the GA approach, as they only differed in their search methods, but not in their definition of the problem.

Having selected our method for mining data, we had to narrow down on a choice of methods for prediction. Our reflection on the reviewed work, described in Section 2.4 on page 23, lead us to three standard approaches: LR, RF, and ANN. Since earlier studies repetitively reported ANNs as having better performance to RF, we pondered if this would be solely due to the ability of ANNs to perform joint feature learning, or if the simplistic approach of majority vote is what hindered RF from performing better.

Naturally, this is a question whose answer would depend on the nature of the problem and features. Thus, we wanted to answer this question experimentally for our particular use case, and give a basis that could be used for other similar studies.

With these choices in hand, we conducted experiments to predict churn, and mine associative rules that connected our signals to the formulated outcome. An overview of the whole analysis process and experiments is depicted in Figure 3.1 on the following page.

3.2 Experiments

There are three main experiments that we designed to for this study. The first one consisted on running prediction models on the data set, to determine how well we could identify churning and retaining customers. The goal of this experiment

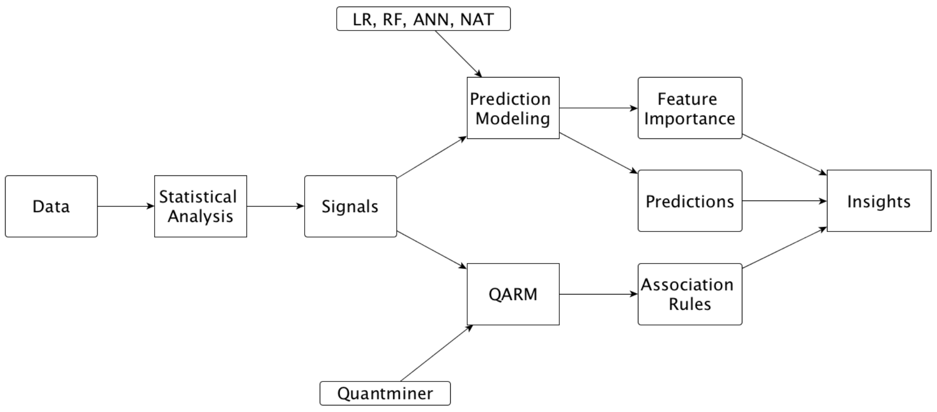


Figure 3.1: Analysis and Experiments Design Process

was to compare different classification models, including our proposed alternate solution for improving classification in a RF. The second experiment was designed to get a measured relevance of the different features we engineered from our data set. In particular, we were interested in seeing what kind of impact each of the feature groups had on prediction performance. For these first two experiments, we made use of seeds in randomization to ensure reproducible results. The third, and final batch of experiments was aimed at extracting quantifiable relations between our signals and the outcome we're trying to predict. This consisted mainly of performing QARM on the data set, having our outcome variable as a guaranteed and only consequent for every generated rule.

3.2.1 Prediction Models

In this section, we describe the configuration of the prediction models that were applied to the data set. Prediction performance is measured on a class and model level. At the class level, we computed precision, recall, and F1-score. At the model level, we compute the weighted average for precision and recall based on the inverse class distribution, as well as the models' F1-score and Receiver Operating Curve (ROC)/Area Under The Curve (AUC). Of particular interest to us is the F1-score measure, which we use to compare performance between models.

Random Forest

The RF model that was used followed the algorithm described in Breiman [64], and is implemented in the python package scikit-learn [65]. This implementation does not directly support categorical features, and so all categorical variables were pre-processed using one-hot encoding.

To tune hyper-parameters we used grid search. The parameters of the grid are detailed in Table 3.1 on the next page. The best configuration was chosen

Table 3.1: Parameters values used in grid search for RF; k represents number of features

Parameter	**Values**
minimum sample leaf	10^2, 10^3, 10^4, 10^5
maximum depth	5, 10, 15, unrestricted
number of trees	25, 50, 75, 100, 200, 300, 500, 1000
maximum features	$\sqrt{k}$, $\log_2 k$

based the following rationale: we looked at error reduction and F1-score increase while observing model stability; then, we picked the first set of parameters after which error would decrease while F1-score hardly changed, meaning, the point in which further improvements to prediction confidence brought no improvements to classification performance.

In addition to this, we wanted a *minimum sample leaf* value that would be meaningful in terms of user coverage, and to keep the number of trees fairly small, as this usually helps when there are correlated features. The best pick was a model with *minimum sample leaf* of 10^3, *number of trees* as 75, and *maximum features* as $\sqrt{k}$, where k is the number of features. *maximum depth* was left unrestricted, as it had no impact on model performance beyond the value of 10. We used 4-fold cross-validation as a method of checking model stability.

One thing we accounted for was randomness in the ensemble, in terms of leaf depth, exemplified in Figure 3.2 on the following page. From it, we can see that even though we left depth unrestricted, the depth of the branches in trees follows a mildly skewed normal distribution, with the mean around 11. The figure also depicts the sample leaf distribution for the same RF model. These visual aids also guided us on hyper-parameter tuning and selection of the number of top features to use for our analysis;

Artificial Neural Network

We used a FFNN. Hyper parameter tuning was done manually, by starting from a small, single layer network, and growing it as needed. It was implemented in the math library tensorflow [66].

Prior to training, all features were scaled using minimum-maximum scaling to the range of $[0, 1]$. Training and testing were split at 80-20%. Similarly here, we used 4-fold cross-validation to check model stability. The find the best model, we kept snapshots at regular intervals during training and recovered the snapshot with the best F1-score on the testing data.

The best architecture had 4 hidden layers, with 16 hidden units in each. The hidden layers were activated using *elu*, as it performed better than *sigmoid*, *relu*, and *tanh*. For the output layer, we used two units, and applied *softmax* for classification. Several optmizers were tested for the cross entropy loss, including *GradientDescent*,

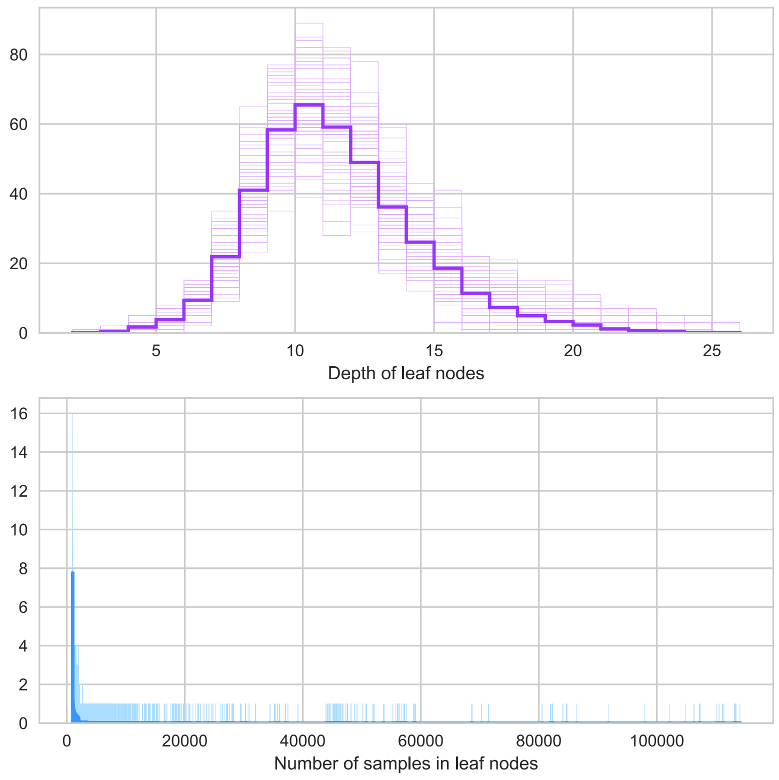


Figure 3.2: Depth and number of samples in leaf nodes: Distribution in RF.

AdamOptimizer, *Adadelta*, *Adagrad*. Between them, *AdamOptimizer* converged the quickest, using a learning rate of 10^{-2}.

We also applied regularization to the model using *dropout*, a technique introduced by Srivastava et al. [67] to prevent overfitting. In theory, using *dropout* is equivalent to training multiple networks with shared weights simultaneously, i.e. an ensemble of networks. It prevents neurons from adapting to neurons from the previous layer, forcing them to learn important patterns. However, when performing inference, all units are active and thus, in the end, only a single model is at work. Best performance was obtained with a *dropout* rate of 30%.

Neural Augmented Trees

To build the NAT model, we picked the best performing RF and took its activation probabilities for the positive class, chosen as activating here, and fed it to a FFNN. We tested two variations of the network: single hidden layer and two hidden layers.

Our intention was to use simple models, as opposed to complex or large ones. This way, we could see if there was value in using this approach as an alternative to feeding the data directly to an ANN. Both networks had 16 hidden units per layer. Once again we used *elu* as the activation function of choice for the hidden units. The output layer had 2 units where we applied *softmax*. We did not apply any *dropout*, and chose the *AdamOptimizer* as the optimizer for the cross entropy loss, with a learning rate of 10^{-2}.

3.2.2 Feature Importance

We measured feature importance at the feature and group level, and we did this by extracting the feature importance information from each individual tree in the RF. To measure the impact of each feature group, we built models that excluded the target group, and then measured changes in F1-score and other metrics in comparison to the model that included all feature groups.

Finally, since feature importance can change in the presence of other features, we extracted feature importance from the model built using all feature groups. This gave us a measure of how features stood out individually to predict churn.

3.2.3 Rule Mining

To perform association rule mining with *QuantMiner*, we needed to define a minimum support and confidence level. Due to software limitations, we could only run the experiment of a relatively small sample of the data, and so to ensure stability for the top rules, as measured by confidence, we ran the algorithm a few times over different sample sets. The formula for rule support is given in Equation (3.1), and the one for confidence is given in Equation (3.2)

$$S = |L \cap R| \tag{3.1}$$

$$C = \frac{|L \cap R|}{|L|} \tag{3.2}$$

L is the antecedent in a rule, while R is the consequent. In our case, we limited R to be our outcome variable churn. *Support* can be measured as a ratio by dividing it by the sample size. We picked 10% to be the minimum *support* as a reasonable threshold for the occurrence of a pattern in the population sample, and 60% for minimum *confidence*. These values were chosen through trial and error, to obtain a reasonable amount of rules that have value, i.e. high enough confidence.

3.3 Data

The data consisted of 1.4 million users, sampled evenly over a period of 14 days. That's 50,000 users for each of the two classes, per day. These limitations were mainly due to the variety of features used, several of which were not available for earlier time periods, and time constraints, as a larger pool of users would require longer data processing time. All users were from a single market, i.e. the United States Of America (USA), to exert some control for population variance.

Each engineered feature was created independently of the others and concurrently for each user, i.e. the computation was data-parallel. A large data set with raw events and defined features was split into chunks, and processed to derive the final feature vector for each user.

Outlier removal was not done for any of the features, since we elected to use summary statistics (e.g. average, median) and ratios as descriptors of user traits. In this process, we carefully tagged each feature with a prefix of their feature group so they could easily be distinguished at a later stage. For missing values, since all features ranged from 0 and above, we used -1 to indicate the absence of data.

After data pre-processing, we ended up with a total of 118 features, split into 4 groups: *demographic*, *activity*, *financial*, and *performance*. *Activity* features described how a user interacts with the service, i.e. engagement; we included features describing temporal aspects of user behavior under this category; the *performance* features described the quality of the experience of the user from the perspective of the platform; *demographic* features related to the structure of geographic populations, with the only two metrics were used in this group being age, and gender identity; finally, *financial* features we used to describe subscription information.

Some of features represented feature vectors of specific traits we were modeling, and the rest were independent. To get to this input vector size, we had to reduce some information by grouping certain distinct features into larger logical groups, and compute derived values from them together. For instance, instead of considering specific product platforms, we merged them into platform types, distinguishing mobile from desktop, and other distinct types. Our justification was that, from our statistical analysis, most features had similar expected value ranges within each platform type. A correlation heatmap of the final 118 features plus the outcome variable is plotted in Figure 3.3 on the next page.

High correlations between some of features could not be avoided, as they stemmed from logically connected, yet distinct signals which we wanted to see how they would fare in relevance. There was a very small number of them compared to the total number of features.

Finally, in many classification problems, there can be class imbalance in the data set. There are several techniques that can be used to address it such as Multiple Period Training Data (MPTD), described in Ali and Arıtürk [27], and Synthetic Minority Over-sampling Technique (SMOTE), described in Chawla et al. [68]. Success rates vary by problem domain, and data features. However, in our case, driven by the availability of data, we opted to use a 50/50 ratio during training, and apply the actual ratio c of churning users during testing. This gave us a $c : (1 - c)$ ratio for churning and activating users respectively. To protect confidential information, we do not reveal these ratios in this report.

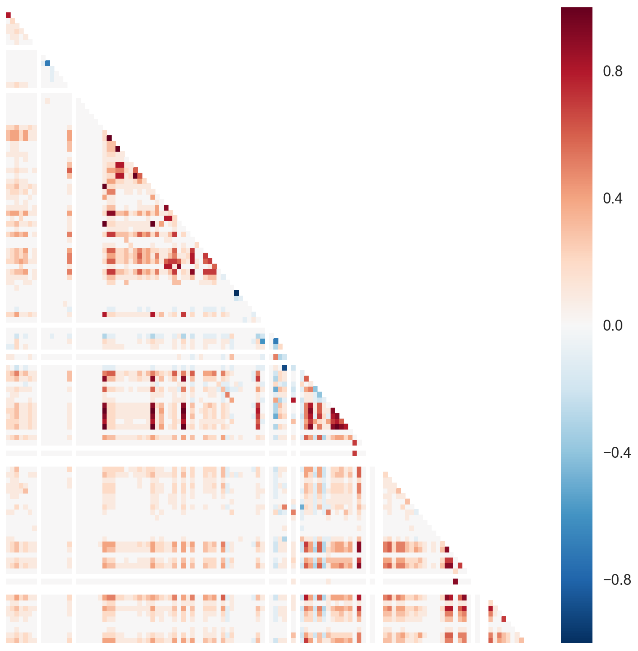


Figure 3.3: Feature Set: Correlation Heatmap

3.4 Activation

As a subscription based service, Spotify AB has users sign up to various subscription plans, with different limitations. Basically, there are two types of plans: paid and free. Currently, a paid plan is either for an individual or a family. These can be organic, in that a user signs up to pay a regular monthly fee to have all the features of the service, or promotional, in that a user signs up through a special package to pay a discounted price for a fixed period of time, e.g. 3 months. For each user that continuously uses the service over a given period of time, e.g. 30 days, we consider them an activating user, i.e. they are active in using the service. Conversely, users that stop using the service after said period are considered churning, even if they are still paying for the service. Alternatively, for users that go from one subscription plan to another, we call them converting users. Our study is concerned with activation of users on the service, as opposed to conversion from one subscription to another.

3.5 Statistical Analysis

Much of the parameters of the study, including the target population, observation and activation windows, and feature selection were defined on the basis of statistical analyses over users, and other logged data. Here, we describe some of the more interesting findings, and how they guided our decision making for the prediction problem.

3.5.1 Population

We began by analyzing users and their churn behaviour with respect to specific subscription plans. We had a study window that went up to 90 days, and for each user in the sample, we observed their subscription migration behaviour. By plotting this using heatmaps and sankeys diagrams, we could study the most frequent subscription transition periods for users. We immediately noticed that users that started on paid subscriptions or full feature trial programs were far more likely to be around after 30 as well as 60 days. However, only a small percentage of the population showed this behaviour. Majority of users actually started on a non-paid plan, and either ended up converting to paid one, or stayed with their subscription. From this, we decided to focus on whether users continued to use the service, as opposed to conversion from one specific plan to another because: (1) usage implies either direct payment or indirect gain through serving ads, and (2) in the beginning it is more important to capture long term engagement from all users, which may lead to conversion to paid subscriptions down the road. Since an overwhelming majority of users signed up through a free plan, we focused our attention on this group, hence our restriction that each user on our sample had spend at least one day on a free plan, to cover the widest range of users of the service as possible. Another observation from our analysis was that the subscription change rate or period differed among users that had distinct traits regarding specific service consumption attributes. These attributes were used at a later stage to engineer informative features for the models.

3.5.2 User Journey

Users in the service engage with the platform with one primary aim: to stream music. However, the ways in which they do this varies from one user to another. This is due to how the product is structured, with different features and sources to stream from. Users can find programmed content, automatically generated content, catalogue content, and content from other users (e.g. user public playlists). As users go from one type of content to another, they have a navigational flow, which we call a *user journey*. We made use of user journey data to understand where users begin, go to, and stream most of their content from. While no explicit journey flow data was used, this analysis helped us derive labels for classifying different types of users based on the source of content. Specifically, we created representation vectors of users, where each factor represented numeric attribution to a given content type. One of the insights of these vectors is how much exploration a user does, as expressed by their streaming source diversity.

3.5.3 Performance Metrics

We aimed to measure if performance metrics, which serve as a proxy to user experience on each platform, would be good indicators of activation. To do this, we ran statistical analysis on the relationship between each of the performance metrics at hand and churn outcome.

There were several performance metrics to chose from, but since we were interested in features that would have a visible, and *felt* impact on users, we narrowed our choices to metrics connected to frequently occurring events, such as playback latency, and metrics with visible impact, like playback errors.

For each of these, we run Kruskall-Wallis tests [69, 70] to see if there was a difference in their values for churning and activating users. For any metric for which the test indicated certain difference, we considered them for the model. This wasn't used, though, as an exclusion criteria, as metrics that failed the test were still considered for the models, though we dedicated less resources into engineering them further.

3.6 Evaluation Metrics

In this section, we outline the metrics used for evaluating the prediction models. Before doing so, we introduce four concepts:

1. TP stands for true positives, i.e. examples that were correctly attributed to a reference class;

2. FP stands for false positives, i.e. examples that are incorrectly attributed to a reference class;

3. TN stands for true negatives, i.e. examples that were correctly not attributed to a reference class;

4. FN stands for false negatives, i.e. examples that were incorrectly not attributed to a reference class.

Precision

Precision essentially answers the questions *What proportion of the examples indicated to be members of a reference class do in fact belong to the class?*. Its value is a fraction of correctly classified examples, over all positively classified examples, and it can be measured for each class individually. See Equation (3.3)

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

Recall

Recall, on the other hand, answers the question *How many of the examples that belong to a reference class were correctly identified.* See Equation (3.4)

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

F1

F1-score is the harmonic mean of precision and recall. Harmonic means are used when we're interested in the average of rates. The metric is commonly used as a measure of overall model performance, both on a class, and model level. See Equation (3.5)

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{3.5}$$

Receiver Operating Characteristic

ROC is a curve used to describe the performance of binary classifiers. It is plotted by placing the false positive rate on the x-axis, and the true positive rate on the y-axis. As a measure, we are generally interested in its AUC.

Intuitively, the AUC of a ROC gives us that probability that given a randomly drawn pair of samples from the positive and negative classes, the positive example will have a higher value in its predicted membership to the positive class. An AUC value of 0.5 indicates that the model performs no better than random chance. The highest value would be 1.0, which would indicate the model can perfectly discriminate all examples [71].

Chapter 4

Results

4.1 Prediction

Results from the prediction models are reported in Table 4.1. The LR model performed the worst, with an F1-score of 50.3%. All the other models, which are non-linear, had significantly better performance. In general, performance variance between them was at most 1% for any given metric, i.e. all non-linear models performed similarly. Both ANN and NAT had mildly higher precision score of 84% compared to the RF model, which had 83.3%. In both cases, this score was at least 34 percentage points higher than that of the LR model. In terms of F1-score, both ANN and NAT with 2 hidden layers had 0.8% advantage over the RF model, while NAT with a single hidden layer had 0.2% lower score, despite both NAT models having had mildly higher ROC/AUC (92.1% and 92.2% for single and double layer networks, respectively).

Table 4.1: Comparison of performance for LR, RF, ANN, and NAT models.

Model	**Precision**	**Recall**	**F1**	**ROC/AUC**
LR	72.3%	58.2%	50.3%	87.6%
RF	83.3%	83.2%	83.2%	91.7%
ANN	84.0%	84.0%	84.0%	92.5%
NAT (1 hidden layer)	84.0%	83.0%	83.0%	92.1%
NAT (2 hidden layers)	84.0%	84.0%	84.0%	92.2%

All models were stable, as observed by their ROC/AUC curves. As an example, we provide the ROC/AUC curve for the RF model in Figure 4.1 on the following page.

4.2 Feature Importance

Feature importance was computed from the decision trees in each RF model. We built models that used all feature groups except one (4 models), and one model

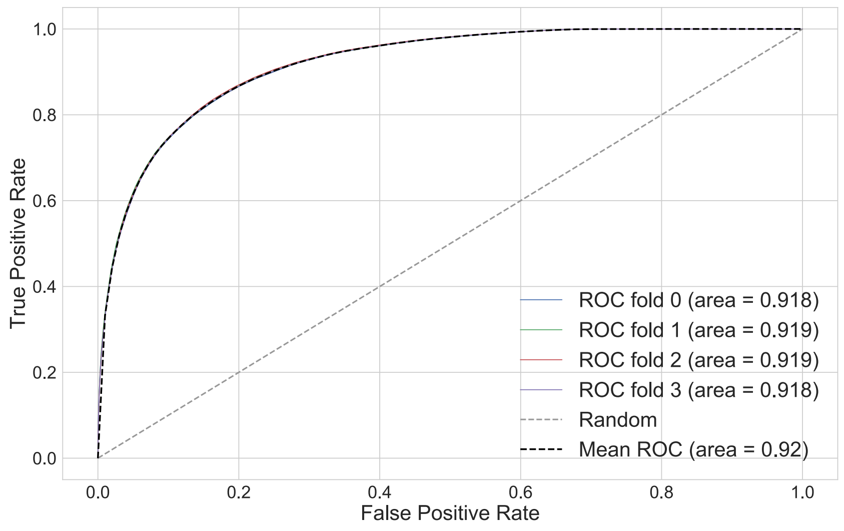


Figure 4.1: Random Forest ROC/AUC curve for 4-fold cross validation. All model lines practically overlap, indicating high model stability.

with all feature groups as the baseline. In total, we built 5 RF models. For each model, with the exception of the model with all feature groups, we discuss the top 15 features.

Table 4.2 details the delta for the weighted average precision and recall, the F1-score, and the ROC/AUC for each of the data group excluding models, while Table 4.3 on the facing page details the delta for precision, recall and F1-score for each of the classes. The only data group whose omission led to a significant drop in performance was the user activity group. Without activity data, recall of activating users dropped by 10.84%, and 2.09% for churners. Conversely, precision for activating users dropped by 3.86%, and 8.93% for churners. F1-score dropped by 7.4%, and 5.55% for the activating and churning classes, respectively. Other models had no significant change to F1-score for each class.

Table 4.2: Global difference between baseline RF model with all feature groups, and specific feature group excluding RF models, measured by percentage points.

Model	**Δ Precision**	**Δ Recall**	**Δ F1**	**Δ ROC/AUC**
No Demographics	0%	0%	0%	-0.01%
No Financial	0%	0%	0%	+0.05%
No Performance	0%	0%	0%	-0.02%
No Activity	-6%	-6%	-6%	-10.07%

For the model without *demographic* data, all top ranking features were activ-

Table 4.3: Per class difference between baseline RF model with all feature groups, and specific feature group excluding RF models, measured by percentage points.

Model	**Class**	**Δ Precision**	**Δ Recall**	**Δ F1**
No Demographics	Activate	-0.08%	+0.08%	0%
	Churn	+0.05%	-0.11%	-0.03%
No Financial	Activate	-0.03%	+0.18%	+0.07%
	Churn	+0.15%	-0.08%	+0.03%
No Performance	Activate	-0.04%	+0.07%	+0.01%
	Churn	+0.05%	-0.06%	-0.01%
No Activity	Activate	-3.86%	-10.84%	-7.4%
	Churn	-8.93%	-2.09%	-5.55%

ity related, with the exception of the 6th, which was *performance* specific. For the *financial* data excluding model, interestingly, more *performance* features were ranked in the top 15, the first of them ranked as 8th, and the other four as 10th, 12th, 13th, and 15th. When we excluded *performance* features, all top 15 features were *activity* related, while model performance remained unchanged. Without activity data though, the average F1-score dropped by 6%, and the most prominent features were *performance* related, up to the 11th, which is followed by *demographic* features. Feature ranking for each of these models is shown in Figure 4.2 on the next page.

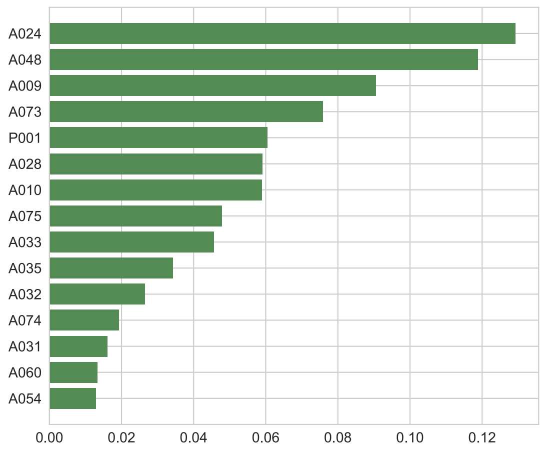
4.3 Mined Behaviour

QuantMiner extracted 743 rules from a sub-sample of the data. All rules had either 1 (44), 2 (124), or 3 terms (575) in the antecedent. Due to limitations of the implementation, we restricted each run to around 200,000 users. To ensure consistency with the top rules, as measured by confidence, we mined multiple samples and observed minor variance in their value bounds, confidence, and support.

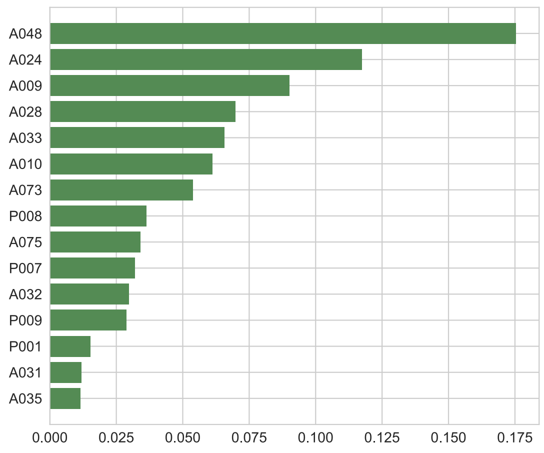
For each rule, we have the frequency, support, confidence. Considering L as the antecedent in a rule, R as the consequent, $\sim L$ and $\sim R$ being their respective logical negations, we have 5 different confidence measures: (1) $L \rightarrow R$, (2) $(\sim L) \rightarrow R$, (3) $R \rightarrow L$, (4) $(\sim R) \rightarrow L$, (5) $L \leftrightarrow R$. We discuss some insights from the top 50 rules.

For one, all of them had a confidence above 90%. Most were related to activity, and temporal features, and almost all involved more than one feature, usually an engineered feature combined with a given feature. From them, we learned value ranges of streaming, and other content specific activity, for which we'd expect a user to activate or churn. Among the mined rules, very few of them pertained to the performance metrics. In fact, only two of them had a significant, and meaningful presence.

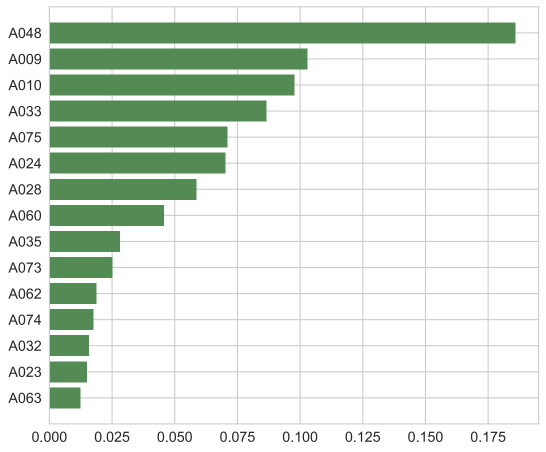
Figure 4.3 on page 41 depicts the distribution of confidence and support of the mined rules. Each each plot, we split them by the number of antecedent terms, and by class, i.e activating and churning. We observe that rules related to churn



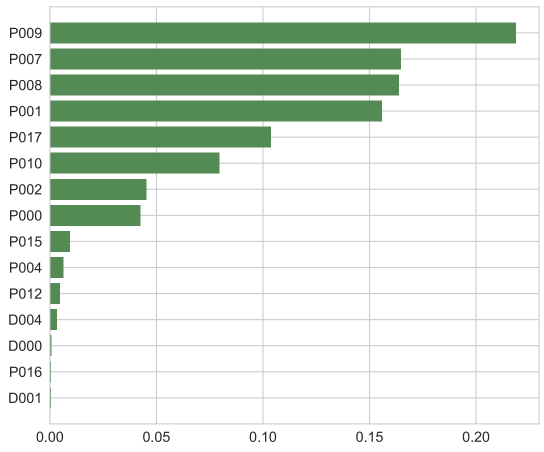
(a) No Demographics



(b) No Financial

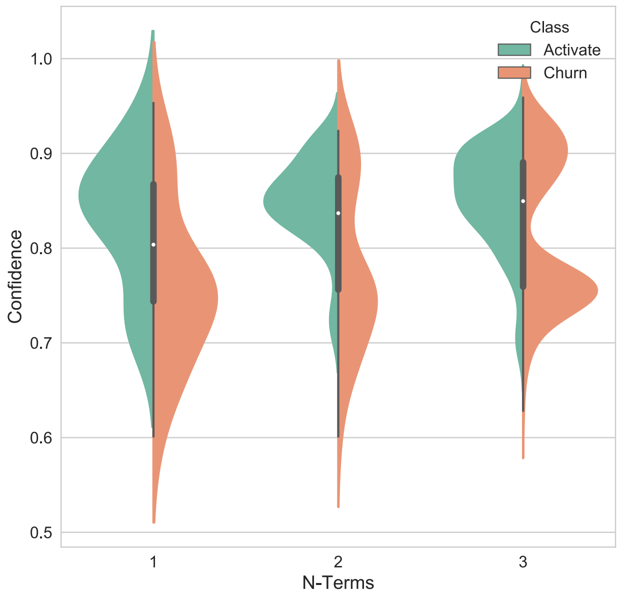


(c) No Performance

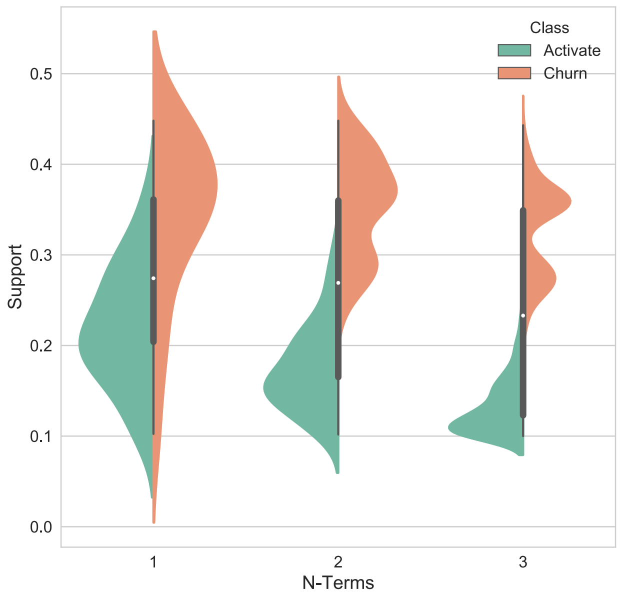


(d) No Activity

Figure 4.2: Top 15 feature ranking for feature group excluding models (y-axis, top-down), measured by importance (x-axis)



(a) Confidence distribution of mined rules, from 0 to 1



(b) Support distribution of mined rules, from 0 to 1

Figure 4.3: Violin plots of confidence and support distribution of 1 (44), 2 (124), and 3-term (575) rules extracted by Quantminer.

patterns have higher support (more prevalent) than activating ones. Confidence for activating patterns, on the other hand, is slightly higher. This is indicative of a trade-off between the two measures; although confidence for both cases was generally high.

Next, we zoom into some aspects of user behaviour hinted by frequency distribution analysis, with particular attention to features that were engineered, and in relationship with the mined association rules. The data pertains to measurements taken during the observation window of the first 7 days, starting from the registration date. To exercise discretion in publishing protected material, the range and domain of values have been transformed. To describe value domains, we use the expression *percentile value points* to refer to quantiles created using all values in the range once, and only once, i.e. a set of unique of elements where non is repeated. So, for instance, if we have a range from 100 to 200, we refer to the 10^{th} value (which is 110) as the 10^{th} percentile value point. The names of features have also been replaced with double letter aliases, e.g. AA, with no particular meaning.

4.3.1 Streaming

We begin by looking at streaming activity related features. First, let us consider streaming feature AA. Figure 4.4 on the next page paints a picture of a long tail distribution of this measure. Churning users have a value lower than the 40^{th} percentile value point of the distribution, i.e. users that have the feature value above this mark are very likely to activate. As we move from the initial point of 0 to higher values, the proportion of activating/churners increases.

Interestingly, when we observe streaming feature AB, it shows a very similar behaviour, but with outliers along the tail (see Figure 4.5 on the facing page). There are users with values in the 70^{th} percentile value point of the range that end up not having any activity on the second week. By carefully observing users, we noticed that some of them end up returning after more than a week. Since our time frame is short, and concerns the first two weeks of a user's experience, we are forced to accept this irregularity for now, for it cannot simply be accounted for with current data. In our rule mining, we found rules that support this trend, with numeric attribution to value ranges, which we could connect to specific platforms and product features.

4.3.2 User Type

A more worked feature is user feature AC. It embodies different aspects of the user's experience, with regards to consumption. Figure 4.6 on page 44 shows the frequency distribution of this measure from churning and activating users. Past the 30^{th} percentile value point mark, users start becoming more likely to activate. This trend grows up to the 60^{th} percentile value point. These observations were

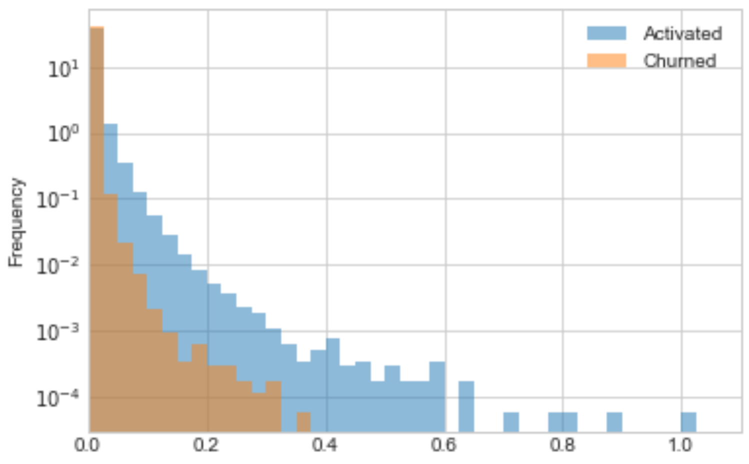


Figure 4.4: Streaming Feature AA Frequency Distribution

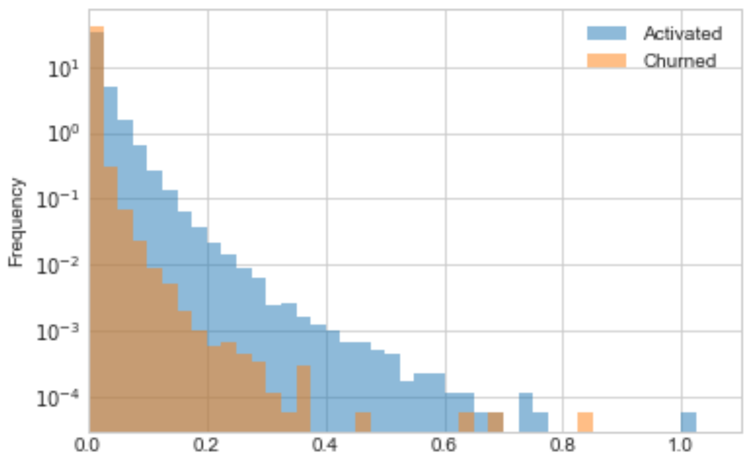


Figure 4.5: Streaming Feature AB Frequency Distribution

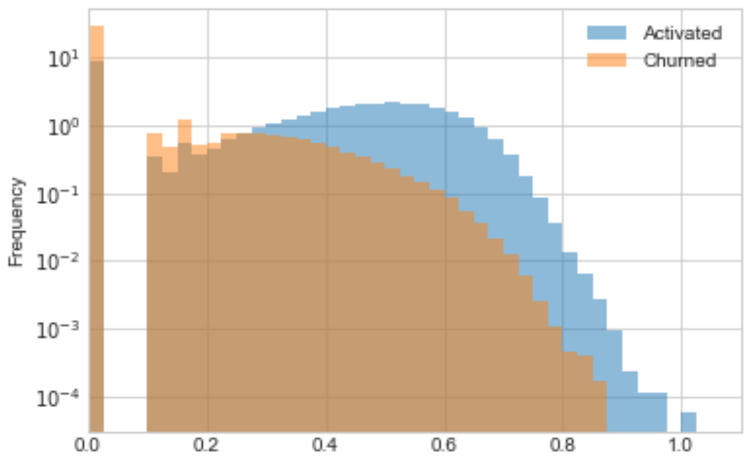


Figure 4.6: User Feature AC Frequency Distribution

supported by mined rules, which indicated high activation probability for users with values past a certain mark.

User feature AD, on the other hand, had a more staircase pattern with steep drops. Throughout the value range, we can see a fairly stable and repetitive pattern in the activation/churn ratio. Above the 50^{th} percentile point, this ratio increases sharply.

User features AC and AD hinted that ratios or distributions of activity could be strong signals for our models, which is why several other features in the different groups were generated using this method. Although combining ratios and volumes can introduce correlations, the potential gains in terms of new joint feature learning from the ANN and the high capacity of RF models made the risk acceptable.

4.3.3 Performance

For one of the performance metrics in study, performance feature AE (Figure 4.8 on the facing page), we observe that its range of values is relatively small, with most values below the 40^{th} percentile value point, and then there are what appear to be outliers. The two final ranges of values leading up to the 40^{th} percentile value point are all of activating users. A possible explanation is that users with values in that range also activate, regardless of how that impacts their experience. Even the first bin, the range with the lowest values, the volume of churning and activating users is roughly the same. As no obvious pattern is visible, we deduce that this feature may not be a very good signal. At least, not on its own, or in a linear space. In our mining experiments, no rules association rules involving this feature were found.

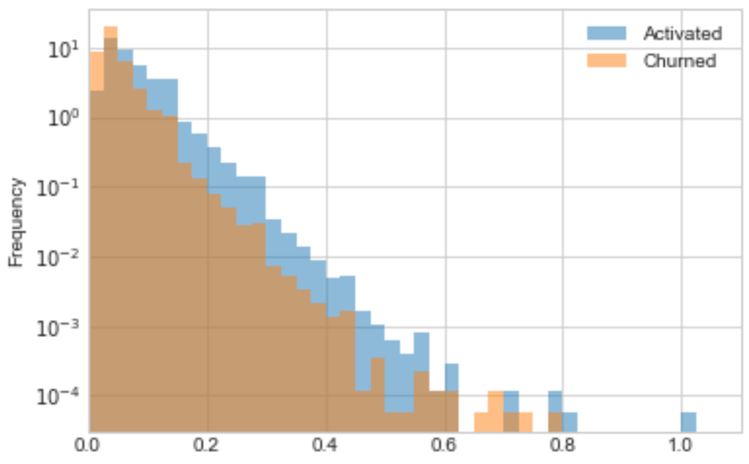


Figure 4.7: User Feature AD Frequency Distribution

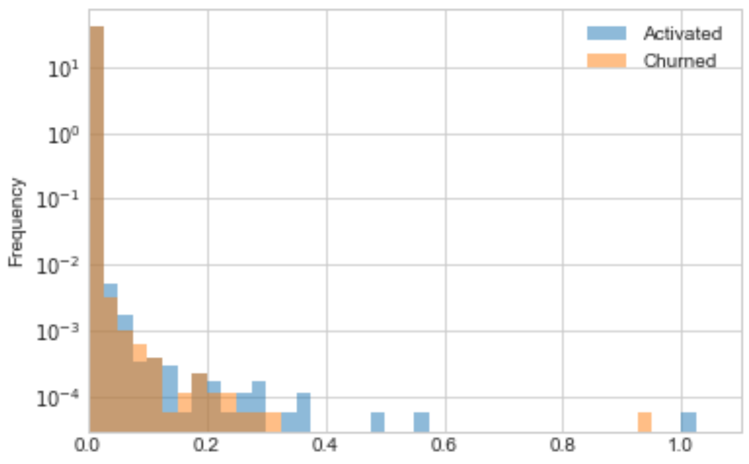


Figure 4.8: Performance Feature AE Frequency Distribution

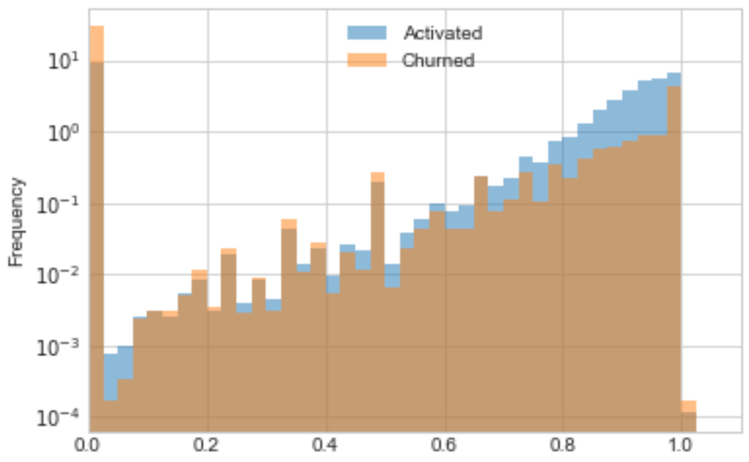


Figure 4.9: Performance Feature AF Frequency Distribution

Performance feature AF (Figure 4.9), on the other hand, shows a slight pattern of correlation with activation. Users that have a value above the 60^{th} percentile value point seem to be more likely to activate. However, this population may be dominated by users with a specific trait, whom also tend to have a higher conversion rate compared to other users. Several mined rules had this feature as an antecedent, indicating its potential value as a performance measure to monitor and observe.

Chapter 5

Discussion

In this chapter we discuss the findings of the experiments that we conducted.

5.1 Prediction Models

The LR model had the lowest performance of all tested models. Particularly, it had a very poor average recall, driven mostly by low recall of churners. Prior to running it, it is hard to anticipate an outcome, so this alone cannot tell us much about the quality of the features or model. However, knowing that each of the three non-linear models tested had significantly higher results while using the same features tells us that there are non-linear relationships which the LR model fails to learn. Hence, our data set requires non-linear models to detect churning users.

We aimed to compare RF, ANN, and NAT. In the end we found that all three performed relatively similar in our experiments. The ANN had the best performance of all three for all measures, including precision, recall and F1-score score, but only by a very small margin. Comparing this performance to the RF model, we can notice an advantage of joint feature learning in ANN over RF. Notwithstanding, the NAT model with two layers performed at par with the ANN. Since with NAT we can still easily extract feature importance and have a slight boost in performance gain, there is value in using a FFNN as an alternative to majority vote in RF in our case. However, it may not outperform a ANN that is directly fed the training data.

As for the features of the models, we determined that while there is information in the performance, financial, and demographics data groups, activity is, perhaps unsurprisingly, by far the most valuable feature group, noting that it also embodies temporal aspects of user's behaviour. What was interesting is that without it, our RF still performed better than the LR that used all feature groups, including activity. So, the non-linear relationships between features also include non-activity related features. This indicated to us that while the non-activity related features are not as informative in the presence of activity related features, they can serve as proxies or direct signals to other user behaviour defining traits.

One factor that might have contributed to the almost equal performance of the three non-linear models we tested is the heavy feature engineering we performed on the data set. Baldi et al. [72] tested building a model to infer composite features based on the input features. They used an ANN, and their model was able to do this, removing the need for them to compute those extra composite features. In our case though, we are not aware of any specific composite relationships between the variables, nor did we test the model by strictly feeding it non-engineered features. It is possible that if we had done so, the ANN might have had significant advantage. This, however, is mere speculation.

Finally, we noticed that by excluding activity related features, the recall drop in activating users was 5 times higher than for churning users, while precision was only half as low, hinting the impact of this feature group in the discovery of activating users and correct classification of churners for our RF model.

5.1.1 A Note on Decision Trees

There are times when one may wish to limit the depth of a decision tree to have short and easily understandable rules. This, however, was not our case. With 118 input features, and several of them being a part of trait vectors, it is not only expected, but also desirable that each DT generates long rules, covering several features at once. This is one of the reasons we opted to use RF models: it gives us a summary of what features are relevant via feature importance, while boosting performance of single decision trees. The only limitation we imposed was on the minimum number of samples per leaf, as way to restrict unnecessary growth of tree branches to cover very small number of users.

5.1.2 Continuous Change

As a constantly evolving and regularly updated service, there can be breaking changes in data logs. There were cases where the meaning of certain fields changed, either due to improvements on semantic interpretation, or because of external factors such as bugs in the applications reporting data. This kind of issue can affect specific versions or revisions of the applications, so we had to filter out the affected revisions for their respective periods. Similarly, throughout the duration of the study, new fields that could add value to the models became available; due to time constraints, most were not considered at all. There were, in total, three iterations of this study, and that was the most we could do with the time in hand.

We note this because, combined with restrictions on data retention, running the same experiments continuously over time, either in a research or production context, may require significant effort in verifying if the meaning or interpretation of features is the same, provided they are still collected, aside from monitoring for model drift.

5.2 Insights

From the studies we conducted, we derived several insights into users and the service itself. Some of the interesting patterns we found pertained to temporal dimensions and product feature usage. Combining these, we can formulate capacity planning, in a way that ensures delivery of an experience that is at par with the standard expected from activating users during specific time periods. We could, also, design experiments that run at specific time periods where users are known to have a more interactive session with a specific type of product feature in the service.

Observing usage, we also learned that certain platforms exhibit more favorable behavior than others, by allowing certain actions. In these, we may see higher engagement from users and higher retention as well. On the other hand, certain behavioural traits found in activating users may be an effect of their choice to continue using the product, as opposed to a cause factor that lead them to decide to keep using the service. In cases such as this, it can be hard to distinguish one from the other. A rule, for instance, could be a mere description of the population.

As an example, consider a hypothetical rule that states that users that stream at least once every hour are 40 times more likely to activate than other users. Logically, someone who streams that frequently is a very active user, and so we would expect them to continue doing so in their second week. Furthermore, it may not be desirable to design an experience that would lead to this behaviour, since not everyone can, should, or would be able to stream at every hour of the day. What this means is that none of the rules can ascertain causality. However, as long as we can find rules that describe traits upon which we can act, or that highlight unexpected aspects of usage, we have gained value from them. This means having rules that pair two unexpected features, or rules that describe a single feature value range that is tied to specific outcome, since these can help us better understand the relationships between product features and user engagement.

As stated earlier, we found rules that connected specific features to a probable churn outcome. Such rules present an opportunity for experiments to find out if the relationship is causal, and if so, which of the antecedents, if any, can serve as a proxy or direct influencing factor.

Conclusively, we found all three non-linear models to be fit at predicting churn for our data set. Though not all rules we mined were valuable, several of them did reveal interesting behaviour signals, upon which we can further explore and understand user behaviour on the service with relation to churn.

5.3 Future Work

Improvement of the work developed in this study can take several dimensions. For one, population control is an aspect that can be better developed and understood for carrying out this kind of study, both by drilling upward and downward, by which we mean comparing results from running the same experiment in different geographic regions (upward) and further narrowing down on population groups based on specific dominant traits (downward), respectively. Perhaps a model designed for a group of users whose main source of consumption vastly differs from the other users could yield better performance, with less complex models, since a general model tries to optimize for the majority of instances in the training sample. The same applies for the rules extracted.

Another avenue of improvement would be in the formulation of the problem itself. While our approach to feature engineering, which was geared at generating a one-dimensional vector for each user, yielded high prediction performance, one could extend it by using a moving window, e.g. on each day, use data for the past seven days to predict the next seven, thus following users as they mature on the platform and measure performance over time; or structuring the problem as a sequential one, where event vectors are generated for each user, perhaps on a daily basis, and use that to predict time to churn. This could be done using models designed for sequential data such as RNN or HMM. These alternative formulations might give different insights on how the different signals help us predict churn and understand users. As for NAT, it would be interesting to run the model on other data sets, and see under what circumstances it can improve a RF classifier, as well as the magnitude of that improvement.

Finally, for the sake of long term value measurement, it would be interesting to run continuous updates to the models, i.e. create a feedback loop. Instances that are correctly classified can be fed back to the model as new input, and those that fail can be investigated to tune the model parameters or features. At the same time, one could run experiments to validate some of the rules generated, and use the outcome to reinforce or discard the rules themselves and, consequently, our understanding of churn drivers.

Appendix A

Software Packages

The following python software packages were used: bokeh (0.12.4), ipython (5.1.0), jupyter (1.0.0), jupyter-client (4.4.0), jupyter-console (5.0.0), jupyter-core (4.2.1), matplotlib (2.0.0), notebook (4.3.1), numpy (1.11.3), pandas (0.19.2), scikit-learn (0.18.1), scipy (0.18.1), seaborn (0.7.1), tensorflow-gpu (1.1.0).

Bibliography

[1] Bruce Ferwerda et al. "Personality Traits Predict Music Taxonomy Preferences". In: *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '15*. Association for Computing Machinery (ACM), 2015. DOI: 10.1145/2702613.2732754. URL: https://doi.org/10.1145%2F2702613.2732754.

[2] Yi-Hsuan Yang and Jen-Yu Liu. "Quantitative Study of Music Listening Behavior in a Social and Affective Context". In: *IEEE Transactions on Multimedia* 15.6 (Oct. 2013), pp. 1304–1315. DOI: 10.1109/tmm.2013.2265078. URL: https://doi.org/10.1109%2Ftmm.2013.2265078.

[3] Noam Tractinsky. "Aesthetics and Apparent Usability: Empirically Assessing Cultural and Methodological Issues". In: *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*. CHI ́97. Atlanta, Georgia, USA: ACM, 1997, pp. 115–122. ISBN: 0-89791-802-9. DOI: 10.1145/258549.258626. URL: http://doi.acm.org/10.1145/258549.258626.

[4] Janette Lehmann et al. "Models of User Engagement". In: *User Modeling, Adaptation, and Personalization: 20th International Conference, UMAP 2012, Montreal, Canada, July 16-20, 2012. Proceedings*. Ed. by Judith Masthoff et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 164–175. ISBN: 978-3-642-31454-4. DOI: 10.1007/978-3-642-31454-4_14. URL: http://dx.doi.org/10.1007/978-3-642-31454-4_14.

[5] John Hadden et al. "Computer assisted customer churn management: State-of-the-art and future trends". In: *Computers & Operations Research* 34.10 (2007), pp. 2902–2917.

[6] Nicolas Glady, Bart Baesens, and Christophe Croux. "Modeling churn using customer lifetime value". In: *European Journal of Operational Research* 197.1 (2009), pp. 402–411.

[7] Ajay Kalia. *"Music was better back then": When do we stop keeping up with popular music? | Skynet & Ebert*. https://skynetandebert.com/2015/04/22/music-was-better-back-then-when-do-we-stop-keeping-up-with-popular-music/. (Accessed on 06/21/2017). Apr. 2015.

[8] Paul Lamere. *Exploring age-specific preferences in listening | Music Machinery.* https://musicmachinery.com/2014/02/13/age-specific-listening/. (Accessed on 06/21/2017). Feb. 2014.

[9] Schwab. *Does Your Risk Tolerance Change Over Time? | Charles Schwab.* http://www.schwab.com/insights/portfolio-management/does-your-risk-tolerance-change-over-time. (Accessed on 06/21/2017). Aug. 2016.

[10] Rob Williams. *Retirement Income Planning: Whatś Your Risk Capacity?* http://www.schwab.com.hk/public/hk/nn/articles/Retirement-Income-Planning-Whats-Your-Risk-Capacity. (Accessed on 06/21/2017). Sept. 2014.

[11] Oliver Smith. "Facebook terms and conditions: Why you dont own your online life". In: *The Telegraph* (2013).

[12] Facebook. *Data Policy.* https://www.facebook.com/about/privacy/. (Accessed on 05/14/2017). 2017.

[13] EC-Justice. *Protection of personal data - European Commission.* http://ec.europa.eu/justice/data-protection/. (Accessed on 05/14/2017). 2015.

[14] Florin Dobrian et al. "Understanding the impact of video quality on user engagement". In: *Communications of the ACM* 56.3 (Mar. 2013), p. 91. DOI: 10.1145/2428556.2428577. URL: https://doi.org/10.1145%2F2428556.2428577.

[15] Molly M. Wasko and Paul M. Di Gangi. "Social Media Engagement Theory: Exploring the Influence of User Engagement on Social Media Usage". In: *J. Organ. End User Comput.* 28.2 (Apr. 2016), pp. 53–73. ISSN: 1546-2234. DOI: 10.4018/JOEUC.2016040104. URL: http://dx.doi.org/10.4018/JOEUC.2016040104.

[16] Heather L OBrien and Elaine G Toms. "What is user engagement? A conceptual framework for defining user engagement with technology". In: *Journal of the American Society for Information Science and Technology* 59.6 (2008), pp. 938–955.

[17] Jan Hartmann, Alistair Sutcliffe, and Antonella De Angeli. "Towards a theory of user judgment of aesthetics and user interface quality". In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 15.4 (2008), p. 15.

[18] Jörg Claussen, Tobias Kretschmer, and Philip Mayrhofer. "The effects of rewarding user engagement: the case of facebook apps". In: *Information Systems Research* 24.1 (2013), pp. 186–200.

[19] Chris Goward. *You should test that: conversion optimization for more leads, sales and profit or the art and science of optimized marketing.* John Wiley & Sons, 2012.

[20] Teck H Ho, Noah Lim, and Colin F Camerer. “Modeling the psychology of consumer and firm behavior with behavioral economics”. In: *Journal of marketing Research* 43.3 (2006), pp. 307–331.

[21] Thanasis Vafeiadis et al. “A comparison of machine learning techniques for customer churn prediction”. In: *Simulation Modelling Practice and Theory* 55 (2015), pp. 1–9.

[22] Abbas Keramati and Ruholla Jafari Marandi. “Addressing churn prediction problem with Meta-heuristic, Machine learning, Neural Network and data mining techniques: a case study of a telecommunication company”. In: *International Journal of Future Computer and Communication* 4.5 (2015), p. 350.

[23] Pierangelo Rothenbuehler et al. “Hidden markov models for churn prediction”. In: *SAI Intelligent Systems Conference (IntelliSys), 2015.* IEEE. 2015, pp. 723–730.

[24] Chih-Fong Tsai and Mao-Yuan Chen. “Variable selection by association rules for customer churn prediction of multimedia on demand”. In: *Expert Systems with Applications* 37.3 (2010), pp. 2006–2015.

[25] Wouter Verbeke et al. “Building comprehensible customer churn prediction models with advanced rule induction techniques”. In: *Expert Systems with Applications* 38.3 (2011), pp. 2354–2364.

[26] Junxiang Lu. “Predicting customer churn in the telecommunications industry—-An application of survival analysis modeling using SAS”. In: *SAS User Group International (SUGI27) Online Proceedings* (2002), pp. 114–27.

[27] Özden Gür Ali and Umut Arıtürk. “Dynamic churn prediction framework with more effective use of rare event data: The case of private banking”. In: *Expert Systems with Applications* 41.17 (2014), pp. 7889–7903.

[28] Dhrubajit Adhikary and Swarup Roy. “Trends in quantitative association rule mining techniques”. In: *Recent Trends in Information Systems (ReTIS), 2015 IEEE 2nd International Conference on.* IEEE. 2015, pp. 126–131.

[29] Ramakrishnan Srikant and Rakesh Agrawal. “Mining quantitative association rules in large relational tables”. In: *Acm Sigmod Record.* Vol. 25. 2. ACM. 1996, pp. 1–12.

[30] Takeshi Fukuda et al. “Mining optimized association rules for numeric attributes”. In: *Proceedings of the fifteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems.* ACM. 1996, pp. 182–191.

[31] Jiuyong Li, Hong Shen, and Rodney Topor. “An adaptive method of numerical attribute merging for quantitative association rule mining”. In: *International Computer Science Conference.* Springer. 1999, pp. 41–50.

[32] Junrui Yang and Zhang Feng. “An effective algorithm for mining quantitative associations based on subspace clustering”. In: *Networking and Digital Society (ICNDS), 2010 2nd International Conference on*. Vol. 1. IEEE. 2010, pp. 175–178.

[33] Gong-Mi Kang et al. “Bipartition techniques for quantitative attributes in association rule mining”. In: *TENCON 2009-2009 IEEE Region 10 Conference*. IEEE. 2009, pp. 1–6.

[34] Anjana Gosain and Maneela Bhugra. “A comprehensive survey of association rules on quantitative data in data mining”. In: *Information & Communication Technologies (ICT), 2013 IEEE Conference on*. IEEE. 2013, pp. 1003–1008.

[35] G Rodriguez. “Poisson models for count data”. In: 2007. Chap. 3.

[36] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M Ingersoll. “An introduction to logistic regression analysis and reporting”. In: *The journal of educational research* 96.1 (2002), pp. 3–14.

[37] Jan Salomon Cramer. “The origins of logistic regression”. In: (2002).

[38] David G Bounds et al. “A multilayer perceptron network for the diagnosis of low back pain”. In: *Proc. 2nd International Joint Conference on Neural Networks*. Vol. 69. 1988, pp. 481–489.

[39] ML Minsky and SA Papert. *Perceptrons–extended edition: an introduction to computational geometry*. 1987.

[40] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks.” In: *Aistats*. Vol. 9. 2010, pp. 249–256.

[41] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

[42] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[43] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. “Fast and accurate deep network learning by exponential linear units (elus)”. In: *arXiv preprint arXiv:1511.07289* (2015).

[44] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks.” In: *ICML (3)* 28 (2013), pp. 1310–1318.

[45] Howard B Demuth et al. *Neural network design*. Martin Hagan, 2014.

[46] Leo Breiman et al. “Classification and regression trees. Wadsworth & Brooks”. In: *Monterey, CA* (1984).

[47] Gareth James et al. *An introduction to statistical learning*. Vol. 6. Springer, 2013.

[48] John Mingers. “An empirical comparison of pruning methods for decision tree induction”. In: *Machine learning* 4.2 (1989), pp. 227–243.

[49] David H Wolpert. “Stacked generalization”. In: *Neural networks* 5.2 (1992), pp. 241–259.

[50] Kai Ming Ting and Ian H Witten. “Stacking bagged and dagged models”. In: (1997).

[51] Saso Džeroski and Bernard Ženko. “Is combining classifiers with stacking better than selecting the best one?” In: *Machine learning* 54.3 (2004), pp. 255–273.

[52] Ibrahim H Osman and James P Kelly. “Meta-heuristics: an overview”. In: *Meta-heuristics*. Springer, 1996, pp. 1–21.

[53] Wai-Ho Au, Keith CC Chan, and Xin Yao. “A novel evolutionary data mining algorithm with applications to churn prediction”. In: *IEEE transactions on evolutionary computation* 7.6 (2003), pp. 532–545.

[54] Ansaf Salleb-Aouissi, Christel Vrain, and Cyril Nortet. “QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules.” In: *IJCAI*. Vol. 7. 2007.

[55] Behrouz Minaei-Bidgoli, R Barmaki, and Mahdi Nasiri. “Mining numerical association rules via multi-objective genetic algorithms”. In: *Information Sciences* 233 (2013), pp. 15–24.

[56] Bilal Alatas and Erhan Akin. “Rough particle swarm optimization and its applications in data mining”. In: *Soft Computing-A Fusion of Foundations, Methodologies and Applications* 12.12 (2008), pp. 1205–1218.

[57] Vahid Beiranvand, Mohamad Mobasher-Kashani, and Azuraliza Abu Bakar. “Multi-objective PSO algorithm for mining numerical association rules without a priori discretization”. In: *Expert Systems with Applications* 41.9 (2014), pp. 4259–4273.

[58] Kalyanmoy Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[59] Yuhui Shi et al. “Particle swarm optimization: developments, applications and resources”. In: *evolutionary computation, 2001. Proceedings of the 2001 Congress on*. Vol. 1. IEEE. 2001, pp. 81–86.

[60] Joannes Vermorel and Mehryar Mohri. “Multi-armed bandit algorithms and empirical evaluation”. In: *ECML*. Vol. 3720. Springer. 2005, pp. 437–448.

[61] Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. “On the Computational Efficiency of Training Neural Networks”. In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 855–863. URL: `http://papers.nips.cc/paper/5267-on-the-computational-efficiency-of-training-neural-networks.pdf`.

[62] Christian Borgelt and Rudolf Kruse. “Induction of association rules: Apriori implementation”. In: *Compstat*. Springer. 2002, pp. 395–400.

[63] J. Mata, J. L. Alvarez, and J. C. Riquelme. “Mining Numeric Association Rules with Genetic Algorithms”. In: *Artificial Neural Nets and Genetic Algorithms: Proceedings of the International Conference in Prague, Czech Republic, 2001*. Ed. by Věra Kůrková et al. Vienna: Springer Vienna, 2001, pp. 264–267. ISBN: 978-3-7091-6230-9. DOI: `10.1007/978-3-7091-6230-9_65`. URL: `https://doi.org/10.1007/978-3-7091-6230-9_65`.

[64] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pp. 5–32.

[65] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[66] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: `http://tensorflow.org/`.

[67] Nitish Srivastava et al. “Dropout: A simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[68] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[69] William H Kruskal and W Allen Wallis. “Use of ranks in one-criterion variance analysis”. In: *Journal of the American statistical Association* 47.260 (1952), pp. 583–621.

[70] Elvar Theodorsson-Norheim. “Kruskal-Wallis test: BASIC computer program to perform nonparametric one-way analysis of variance and multiple comparisons on ranks of several independent samples”. In: *Computer methods and programs in biomedicine* 23.1 (1986), pp. 57–62.

[71] James A Hanley and Barbara J McNeil. “The meaning and use of the area under a receiver operating characteristic (ROC) curve.” In: *Radiology* 143.1 (1982), pp. 29–36.

[72] P Baldi et al. “Searching for exotic particles in high-energy physics with deep learning. Nat. Commun”. In: (2014).

Declaration

I hereby certify that I have written this thesis independently and have only used the specified sources and resources indicated in the bibliography.

Stockholm, September 3, 2017

Guilherme Dinis Jr.

. . . .

Guilherme Dinis Chaliane Junior

TRITA Trita-ICT-EX-2017:158