INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Social network Embeddings for Churn Prediction

Hugo Filipe Paulino dos Santos

Mestrado em Informática e Gestão

Orientador(a):
Doutor João Pedro Afonso Oliveira da Silva, Diretor de Informática,
ISCTE

Co-Orientador(a):
Doutor Miguel Ângelo Leal da Costa, Lead Data Scientist, Vodafone

Novembro, 2020

"Never discourage anyone... who continually makes progress, no matter how slow."

Plato

Resumo

Com a generalização da Internet os clientes tornaram-se mais informados dos serviços existentes e dos seus preços. Na perspetiva das empresas, adquirir um novo cliente é mais dispendioso que manter os existentes. Nesse sentido as empresas começaram a abordar o desafio da saída de clientes para outras companhias. A saída de clientes é ainda mais desafiante no setor das telecomunicações, porque os clientes podem mudar de operador com maior rapidez devido ao período de fidelização mais curto e à fácil migração do serviço para outros operadores de telecomunicações sem custos associados. Antecipar a saída é, portanto, uma grande preocupação para as empresas de telecomunicações, que as leva a realizar campanhas de retenção para esses clientes.

Modelos preditivos permitem prever se um cliente vai abandonar a sua operadora atual usando informação passada desse cliente. O presente trabalho detalha como foi construído um modelo preditivo para prever a saída de clientes explorando relacionamentos entre clientes. Ao contrário de outros trabalhos, este utiliza uma análise de rede social que tira partido de representações de baixa dimensionalidade dos clientes (network embeddings) e permite obter melhores resultados que outros métodos.

Paravras-chave: Rotatividade de clientes, Análise de redes sociais, Representações latente de rede.

Abstract

With the large adoption of Internet customers became more aware of existing services and their prices. From the perspective of companies acquiring a new customer is more expensive than maintaining existing ones. In this sense, companies began to address the challenge of leaving customers to other companies. Customer churn is even more challenging in the telecommunications sector, because customers can change operator faster due to shorter loyalty period and easy migration service to other telecommunications operators without associated costs. Anticipating churn is therefore a major concern for telecommunication companies, which leads them to carry out retention campaigns for these customers.

Predictive models allows us to predict whether a customer will leave their operator using that client's past information. The present work describes how a predictive model was build to predict the outflow of customers exploring customer relationships. Unlike other works, it uses a social network analysis that takes advantage of small customer representations (network embeddings) and allows to obtain better results than other methods.

Keywords: Customer Churn, Social Network Analysis, Network Embeddings.

Acknowledgements

First, I would like to thank my supervisors, Professor João Oliveira and Miguel Costa who contribute with knowledge, scientific input and suggestions that made this work possible and who have shaped the approach to this result. I am very grateful for their support and patience.

To my mother, sister and friends who supported me all times and are always there when needed.

Table of Contents

List of Figures

List of Tables

Acronyms

CDR	**C**all **D**etail **R**ecords
ML	**M**achine **L**earning
SNA	**S**ocial **N**etwork **A**nalysis
NE	**N**etwork **E**mbeddings
AUC	**A**rea **U**nder **C**urve
ROC	**R**eceiver **O**perating **C**haracteristics

Chapter 1

Introduction

We live in a world with more and more information. With the increasing use of new technologies, customers are now much more informed about products, services and companies. Customers usually look for better services, and at a better price. This makes companies increasingly concerned about keeping up to date with their products and evolving them, because acquiring a new customer costs much more than retaining an existing customer [2, 45, 42, 21].

Companies began to address the churn problem. Churn is the migration of a customer to another company or service. This problem is transversal to all economic sectors such as: financial, telecommunications, insurance and healthcare. Predicting when a particular customer will leave requires understanding what motivates customers to stay and which services the customer is most interested in. In this sense customer information is crucial to create a good churn forecasting system.

Previous work has shown that the influence of people may lead to churn [16, 12, 44]. For instance, if all my friends leave to another telecommunication's company, I am more likely to leave as well. This thesis explores the use of social networks analysis applied to churn, which means using data from customers and their relationships to predicting whether a particular customer will leave the company.

Social networks are an important class of networks that can represent, connections between people, telephone caller-callee, citation of academic papers and many others [24].

In the context of telecommunications a social network can be represented by a graph, where customers are the nodes and interactions between customers the links/edges. This network can became large and sparse, since each call, sms made is an interaction between two customers. In addition it is computationally expensive to use this information in churn prediction models. This leads that most of the churn prediction models created, ignore relational information about customer relationships. Another limitation of using social networking analysis is that it only consider first degree relationships, for example if you have several friends, only these will be considered for churn prediction, (It does not take into account friends of my friends) Figure 2.1 represents an example of a social network in telecommunication sector. Initial customers are represented inside the circle G = 1, G = 2 represents the first line connections and G = 3 represents the second line connections.

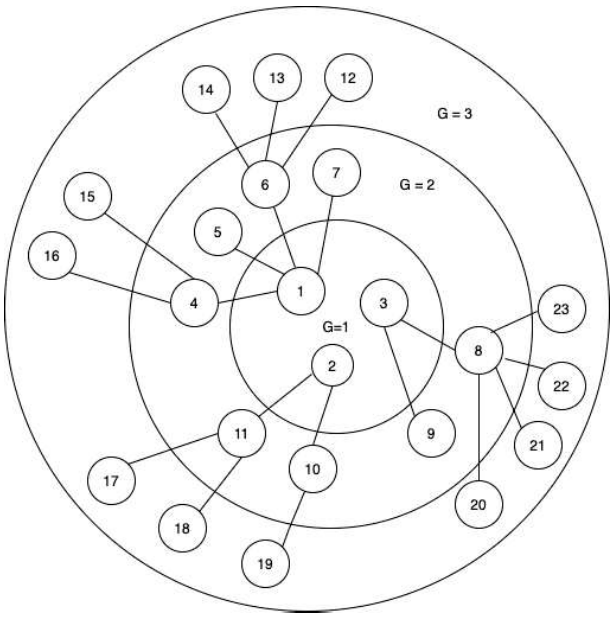


Figure 1.1: CDR Graph with different type of connections.

Representation learning allow us to represent social networks in a way that is more easy to use [31]. Representation learning focuses on obtaining a vectorial representation such that graph analysis become easy to do. This view is particularly helpful for processing structured data, i.e. sequences, trees, and graphs, where vectorial representations are not immediately available.

Representation learning allows to constructing dense representations in this case (Network

Embeddings) of objects in a low dimensional space. Network embeddings create representation of nodes as vectors where similar nodes are closer to each other, for instance, in an euclidean space [8]. These representations can then be used as features for common tasks such as classification, clustering, link prediction and visualization.

In this thesis we explore how connections between people and the use of Network Embeddings can help predicting churn.

1.1 Motivation

Churn forecasting is an important topic in sectors such as telecommunications [20, 44, 12], finance [42, 27, 48], insurance [15, 38, 13]. There are many related works in the area, however most of them focus on the customer individually, while just a few focus on customer relationships. Studies demonstrate that we can have a more precise model using the relationships between customers [33]. Using social network analysis (SNA) we are able to discover relationships between customers, however most of the existing algorithms only rely on first order connections between customers, i.e, only detects the "friends that I call most", discarding the "friends of my friends". Representation learning (Network embeddings) overcome this limitation of discover relationships and are able to discover relationships between customers from the first order to n order connections. In this sense, our motivation is to create a churn prediction model that uses customer relationships to predict whether a customer will leave the company using representation learning (Network Embeddings).

1.2 Goals

In this dissertation we intend to take advantage of data available in telecommunication companies to improve churn prediction. Studies demonstrate that using social data (communications between customers) in telecommunication churn models increase the performance of the existing ones, we pretend to use social network analysis to create a model that predicts churn. We intend to use network embeddings, to improve the existing churn prediction models. Network embeddings is a representation learning technique that converts

data in a graph to a vectorial way. The goal of this thesis is to develop a churn prediction system using interactions between customers (calls made to other customers and respective duration) and whether representation learning can improve the performance of the existing churn prediction system.

1.3 Contributions

The major contributions of this work are the following:

- We purpose a novel method to detect churn in telecommunication sector.
- We perform a experimental evaluation in one of the biggest telecommunication operators in the world.
- We provide guidelines for other researchers or business managers use the potential of these techniques.

1.4 Thesis Outline

The rest of this dissertation is organized as follows:

Chapter 2 (Related Work) provides background information and related work that are needed to understand this dissertation;

Chapter 3 (Churn Predictive Modeling) introduces the pipeline and solution to create our churn prediction system;

Chapter 4 (Results) presents the results of the techniques applied in order to get a good insight about the quality of the models and validate our thesis;

Chapter 5 (Conclusion and Future Work) concludes this dissertation and discusses future improvements;

Chapter 2

Related Work

This chapter presents an overview of the state of the art in several topics needed to understand this thesis work. It starts by describing customer churn in multiple business sectors, such as Finance, Insurance and Telecommunications in Section 2.1 (Customer Churn). In Section 2.2 (Social Network Analysis), besides an explanation about SNA, data used for Social Network Analysis and advantages of using SNA in churn prediction is described. Finally Section 2.3 (Network Embeddings) describes Network Embeddings and how it can overcome the limits over Social Network Analysis using graphs.

2.1 Customer Churn

The Internet increased the awareness of services and products provided by companies. This in turn, increases customer churn.

Customer Churn is defined as customer migration from one company to another.

Companies invest a lot of time and money in trying to solve this problem, since acquiring a new customer has a much higher cost than retaining an existing one [42, 21].

Lazarov and Capota [23] defined the following type of customers:

Voluntary churn: Customers who decided to leave the service provider to another one. The reasons for this are associated with: dissatisfaction with the service (i.e bad internet quality), bad support (call center line), no rewards for customer loyalty, etc.

Rotational/Incidental churn: Customers quits contract without the aim of switch to a competitor. Reasons for this are changes of the circumstances, for instance change of geographical location of the customer to a place where the company is not present or not has the same service. Financial problems leading to impossibly of payment.

Non voluntary churn: Customers to whom the company closed their services (e.g. because of missing payment).

Predicting Voluntary and Rotational churn is therefore a major concern in companies, since we want to predict beforehand whether a customer will leave the company. Churn is not a problem of a particular economic sector, thus to better understand its similarities and differences, we present the state of the art for several economic sectors in the following sections. At the end a summary is shown.

2.1.1 Churn in Finance

Mutanen et al. [27] affirmed that customers stay with a company for very long time, i.e. they do not switch their financial provider often. In their work they predicted churners using logistic regression algorithms and used lift curves as evaluation method. Xie et al. [48] tried a new approach for churn prediction using Imbalanced Random Forests and compared the performance against artificial neural network (ANN), decision tree (DT), and class-weighted core support vector machines (CWC-SVM). Van den Poel and Bart Lariviere [42] used customer behavior, customer demographics and macro-environment to predict churn in finance. In their approach they used Survival Analysis and proportional Hazard Modeling to detect customer churn over time. They found that individuals experienced an high switching probability in the early years after becoming a customer. After 7 years, the likelihood to stay with the company stabilizes for a period of 15 years. Then, after 20 years as bank customer, the probability to staying with the company decreases.

2.1.2 Churn in Insurance

Guillén, Montserrat and Nielsen et al. [15] studied the time varying factors which explain the customer lifetime duration. The authors identified the customers with a higher risk

of cancelling contracts and how that risk evolves over time using proportional Hazard regression models. Soeini et al. [38] used another approach with data from seven insurance companies. They applied K-means to cluster them based on their similarities and a Decision tree algorithm was applied at each cluster. In the same line of work, Goonetilleke et al. [13] predicted churn using decision trees and neural networks.

2.1.3 Churn in Telecommunications

The Telecommunication sector is where subscribers are more dynamic, which means that they change provider more easily [30], unlike other sectors such as finance and insurance where the customers loyalty increases over time [42].

Hung et al. [21] demonstrated the importance of customer churn in the telecommunication sector and explored which features are the best for their models. The data used consists in a population of 160.000 customers with 0.71% of churn rate in a Tawian operator. To detect the best features for his models, they review other research papers and interviews with telecom experts. After that they have created two experiments, in the first one K-means was used to segment the customers into five clusters based on customers similarities, Decision Tree algorithm C5.0 was applied at each cluster. The second experience they apply Neural Network algorithm without clustering. Lift curves and hit ratio are used for testing the performance of the models. His findings demonstrate stable accuracy in the first 6 months, Neural Network algorithm outperforms Decision Tree.

The study of Neslin et al. [28] is based on a tournament in which both academics and practitioners participate. The Data used was publicly available and consists in 100.000 customers with a 1.8% of churn rate. The features used in this challenge to predict churn are not detailed. The participants tried multiple techniques for predicting customer churn such as Logistic Regression (45%), decision trees (23%), NN (11%) and oversampling to increase algorithms accuracy. The results suggests that predictive models stays stable for a period of approximated of three months.

Xia et al. [47] compared the performance of SVM kernels against other algorithms like ANN, C4.5, Logistic Regression and Naive Bayes.To test his models they used two public datasets the first one with 3333 samples, for training 2850 non churners and 483 churners

and kept 1667 samples for testing. Dataset two contains 1443 non churner and 224 churner customers. Their conclusions demonstrate that less missing data and abundant attributes that use with SVM lead to a good prediction precision.

Tsai et al. [41] describes two hybrid experiments for churn prediction, the first one combines clustering with classification techniques, i.e., they have applied clustering, after that a artificial neural network was applied at each cluster. The second one combines two classification techniques. At first they apply a ANN to the dataset that gives one output and then they apply ANN again. Fuzzy testing data (FDT) was also applied, which was based on filtered out data by the first model of the two hybrid models. The data used was provided by an American telecom dataset which contains 51306 subscribers, 34761 churners and 16545 non churners They found that using hybrid models outperform the baseline model with artificial neural networks.

In Owczarczuk [30] work they compared the performance of Logistic regression against decision trees in prepaid segment. The dataset was provided by a Poland mobile operator which contains 1381 variables 85.274 samples 36.824 churners and 45.497 non churners. Since it was computationally expensive to use 1381 attributes for training, he decided to use the t-test to retrieve the 50 most important ones. His conclusions demonstrate that linear models are more stable than decision trees.

Huang et al. [18] described a new approach to select the best features to predict churn. To select the best features for the algorithm they modify a genetic algorithm called NSGA-II. After this process the a decision tree perform the classification on those features. The dataset used was from Ireland telecom Eircom it contains 23.600 samples with 18000 non churners and 5600 churners.

Verbeke et al. [45] used rule-based models to understand which features leads to churn. They have used AntMiner+ and ALBA models and compare it against logistic regression, C4.5, RIPPER, and SVM. The used a public dataset with 5000 samples and 21 attributes where, 14.3% are indicated to churned. They have showed that ALBA combined with RIPPER and C4.5 gives the highest accuracy. AntMiner+ allow us to include domain knowledge and results in most comprehensible models.

Huang et al. [19] focused on the usage of more features to improve churn prediction models. Seven models (Logistic Regression, Naive Bayes, Linear Classifiers, Decision Tree C4.5,

Support Vector Machine, Neural Network, Data Mining by Evolutionary Learning) were used and compared. They also performed feature normalization for some predictors (e.g. Neural Networks). They used a dataset with 827124 samples containing 13562 churners and 40000 non churners, the dataset has 738 attributes. The techniques such as SVM and Decision tree C4.5 perform better. DMEL and Naïve Bayes doesn't perform well with a large number of features.

Verbeke et al. [43] proposed a new metric to get the most profit in customer churn optimizing the maximum profit that can be generated by a marketing campaign. A large benchmark was conducted including twenty one predictive algorithms over eleven telecom datasets. They argue that applying the maximum profit criterion and including the optimal fraction of customers in a retention campaign leads to substantial different outcomes.

Ballings and Van del Poel [3], applied logistic regression, classification trees in combination with bagging to study the relationship between the length of customer event history and classification performance. They conclude that the length of the predictors period is logarithmically related to classification performance.

2.1.4 Summary

The previous studies address customer churn in Finance, Insurance and Telecommunications. We can conclude that the algorithm most used in churn prediction are Logistic Regression, although most of the works considered using more than one algorithm. Usually the data used in churn prediction are imbalanced, which means that the percentage of churners is always low. This can be a problem for less sensitive algorithms to generalize. There are little information about the features used for churn prediction in the presented works. Most of the algorithms presented focus on the customer individually, this is, churn is predicted having into account only the customer and his attributes. In the next chapter we review the use of Social Network Analysis for customer churn prediction in telecommunication sector.

2.2 Social Network Analysis

Social network analysis is the process of investigating social structures through the use of network and graph theory [29]. It characterizes network structures in terms of nodes (individual actors, people, or things within the network) and the ties, edges, or links (relationships or interactions) that connect them. A graph, G consists in a set of V, vertices denoted as nodes and a E, collection of pairs of V called edges. Edges are represented by **(u,v)** where **u** represents the origin and **v** represents the destination and are either Directed or Undirected. Directed when edges have a direction. The edges indicate a one-way relationship. Undirected when edges do not have a direction. The edges indicate a two-way relationship.

In order to understand social network analysis we first need understand how we create a network using data from telecommunications.

2.2.1 Telecommunication Data

In Telecommunication sector we have data about the customers, such as: calls made to other numbers, information about the customer, information about payments, and customer service information.

We can have two segments:

Prepaid: The customer does not have a subscription or contract associated. It is more challenging to predict churn in this segment since we have less information about the customer [30].

Post-paid: The customer has a subscription associated. Postpaid customers usually churn at the end of the contract, which makes it easier to predict [30].

In Prepaid segment we usually create data features about mobile usage extracted from Call Detail Records (CDR) such: Inbound Calls, Outbound Calls, SMS send, SMS received and Internet usage.

In Postpaid segment we have contract information about the customer. Usually the customer creates a subscription from month **n** to **m** and pays a monthly fee for the service. In this segment providers have more customer data available, including:

- **Demographic information:** Age, sex, education level, martial status, occupation;
- **Billing information:** Monthly charges, products subscribed, total charges, payment method, contract;
- **Customer Service information:** If phone number was changed or suspended, and contact center information about customer;

A more detailed description of the previous works using those features can be found in Table 2.1.

Authors	Feature types	Description	Segment
[21], [6]	Demography	Gender, Age, Area Tenure	PostPaid
[21], [36], [44]	CDR	Inbound calls, Outbound calls, Domestic call, SMS	Prepaid and Postpaid
[21], [6]	Bill/payment	Bill amount, Payment, Overdue payment, Monthly fee	PostPaid
[21]	Customer Service	Inquire, Phone no changed, Bar/suspend	PostPaid

Table 2.1: Types of features typically used to predict churn.

2.2.2 Churn Prediction using SNA

In Telecommunication sector we can represent the data using a directed graph which means the edges have indicate one-way relationship, the nodes denote the clients and edges the interactions between them (Inbound calls, Outbound Calls, SMS received, SMS send).

We can see a example of a Social Network on Figure 2.1. It represents a call network graph with multiple levels: *G=1* contains initial customers first order connections (directed interactions) between *G=2*, G=2 contains second order connections between *G=3*.

Related works using Social Network Analysis are summarized in the Table 2.2.

Dasgupta et al. [12] created and analyzed customer churn using social ties between the subscribers of a telco operator. To the best of our knowledge this work is the first of its kind. The approach is focused on prepaid customers. A CDR graph was created by linking customers as nodes and the interactions between them as edges. They developed a diffused based modeling technique called Spreading Activation which is based on the premise that few key individuals (churners) may lead to strong word of mouth effects wherein they influence their friends to churn. At the end of spreading activation each

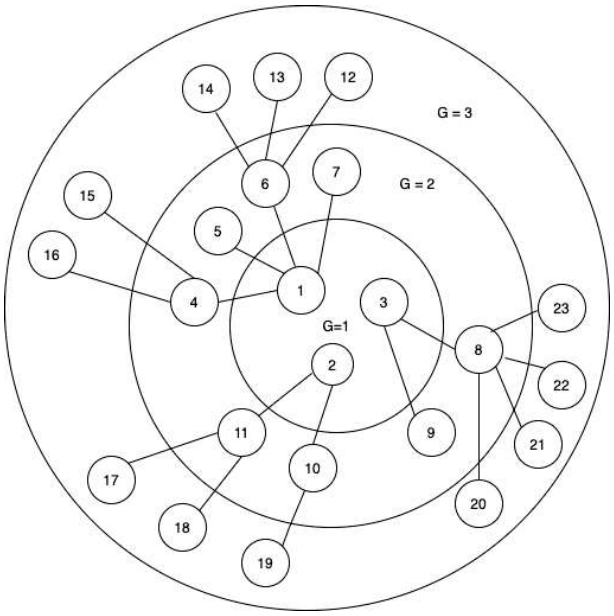


Figure 2.1: CDR Graph with different type of connections.

node has a probability of churn. Their study confirms the relevance of social network information in predicting churn.

Authors	Algorithms	Dataset	Year	Metrics
Dasgupta [12]	Spread Activation (SPA)	CDR Data(Calls and SMS)	2008	Lift Curves
Richter [34]	Random Walks, Markov Chains, Decision tree	28 days of CDR Data(Calls)	2010	Lift Curve
Saravan [36]	Graph Parameter, Analysis, Logistic Regression	3 Datasets(3 month record)	2012	T-test
Verbeke [44]	Graph Parameters, Logistic Regression	5 months of CDRs	2014	Lift and MPC(Maximum Profit Measure)
Huuang [20]	Page Rank, Label Propagation	9 Months dataset	2015	AUC
Ahmad [1]	PageRank, Sender Rank, Degrees,Similarities, Betweenness	4 months CDR	2019	AUC

Table 2.2: Related works in churn prediction using Social Network Analysis.

Ritcher et al. [34] implemented a churn predicting approach which computes the churn probability by initially calculating group churn scores and then obtaining individual churn scores based on social rank within their group. They have used CDR data to detect social groups and communities. In these groups there is a leader which is the customer who has the most influence over the group. They found that when a leader is a member of a competitor network it has an important effect on churn. The person most likely to churn in the group is the leader, three times more than the other customers. Being a part of a larger group makes churn less likely.

Verbeke et al. [44] created two experiments using CDR. First created a graph to detect the social impact of churners mixed with a non-relational classifier approach. Non relational

classifier approach means predict churn individually, without having into account relationships between customers. They have shown that using social network relational classifiers does not detect the same churners as the non-relational classifiers. However using the two strategies can improve the profit in the company because it detects different churners.

For evaluation metrics most the previous works used Lift Curve. Only Ahmad et al. and Huang et al. used AUC [1], [20].

The previous works using SNA have the following limitations:

Label propagation: only influence propagation from directed connections are considered. Influence (e.g. churn influence) propagation from n-order (n>1) nodes are ignored. For instance, in Figure 2.1 the nodes in G=3 do not influence nodes of G=1.

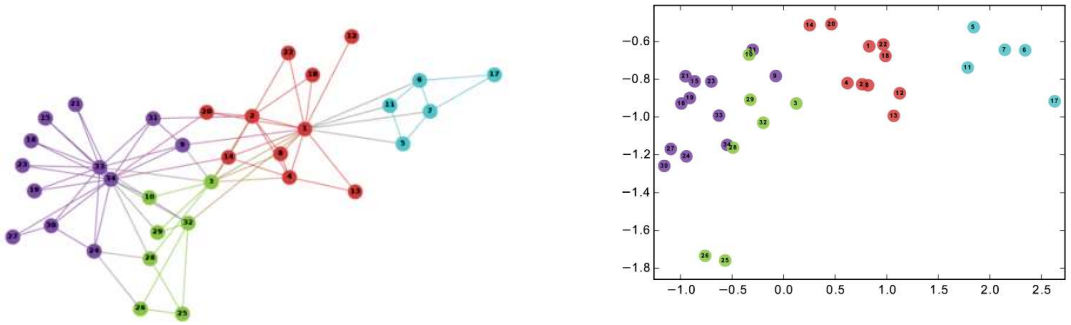
Node Representation: Represent all the nodes in the graph. This limitation is related to the extraction of structural information traditional approaches rely on graph statistics such as degrees or clustering coefficients, or carefully engineered features to measure the local neighborhood. This approaches are inflexible, they not adapt during the learning process.

To overcome these limitations Network Embeddings are proposed in the next chapter Chapter 2.3 (Network Embeddings).

2.3 Network Embeddings

Network embeddings were proposed to overcome the complex inference procedures over graph networks with billions of nodes and edges. The central idea is to find a mapping function which converts each node in the network to a low-dimensional latent representation. These representations can then be used as features for common tasks on graphs such as classification, clustering, link prediction and visualization [8]. **Definition Network Embeddings**: For a given a network G, a **network embedding** is a mapping function $\Phi : V \mapsto \mathbb{R}^{|V| \times d}$. This mapping Φ defines the latent representation of each node $v \in \boldsymbol{V}$.

An example of network embeddings is depicted in Figure 2.2. The Figure 2.2 (a) represents a graph of a social network. In Figure 2.2 (b) one network embedding representation of graph (in this case 2-dimensional representation).



(a) A graph representing a social network

(b) A continuous space embedding of the nodes in the graph.

Figure 2.2: Example of network embeddings [8].

As previously discussed in Section 2.2 (Social Network Analysis) we point some limitations on Social Network Analysis. One of those limitations is the scalability of networks that contain billions even trillions of nodes. To solve the scalability issue we need to reduce the network dimensions. Multiple techniques were proposed to reduce graph dimensionality such: Principal Component Analysis (PCA)[46], Multidimensional scaling (MDS)[22], Isomap[40], Local Linear Embeddings (LLE)[35] and Laplacian eigenmaps (LE)[4]. However the performance of the above methods have been shown lower performance compared to network-based approaches since the complexity of these methods are at least quadratic which makes it impossible to run in large networks [8]. With the age of deep learning new methods arise such: DeepWalk[32], Node2vec[14], LINE[39], GraRep[7].

2.3.1 Preliminary Concepts

Before we dive into network embeddings algorithms we later introduce some concepts first. We will describe those concepts and after we explain some algorithms such as: DeepWalk, Node2Vec, LINE and GraRep.

First order proximity - First-order proximity is the local pairwise proximity between two connected vertices [39]. For each vertex pair (v_i, v_j),if $(v_i, v_j) \in E$, the first-order proximity between v_i and v_j is w_{ij}; otherwise, the first-order proximity between v_i and v_j is 0. The first-order proximity captures the direct neighbor relationships between vertices.

Second order and high order proximity - The second-order proximity captures the

2-step relations between each pair of vertices [39]. For each vertex pair (v_i, v_j), the second order proximity is determined by the number of common neighbors shared by the two vertices, which can also be measured by the 2-step transition probability from v_i to v_j equivalently. Compared with the second-order proximity, the high order proximity captures more global structure, which explores k-step ($k \geq 3$) relations between each pair of vertices. For each vertex pair (v_i, v_j), the higher order proximity is measured by the k-step ($k \geq 3$) transition probability from vertex v_i to vertex v_j , which can also be reflected by the number of k-step ($k \geq 3$) paths from v_i to v_j . The second-order and high-order proximity capture the similarity between a pair of, indirectly connected, vertices with similar structural contexts. We can only have orders of more than two using network embeddings. Network embeddings allow us to detect hidden connections in most of the cases where we have second and high order proximity in our graph. An example of this can be found in Figure 2.1

Random Walks - Random walks is an algorithm that goes through a node in a random set of a directions. At each step the algorithm moves on to a new random direction in the graph until it reaches a maximum length. It's goal is to capture structural relationships between vertices. By performing truncated random walks, an information network is transformed into a collection of vertex sequences, in which, the occurrence frequency of a vertex-context pair measures the structural distance between them. Vertex representations are then learned by using each vertex to predict its contexts. An example is presented in Figure 2.3

SkipGram - SkipGram is a language model that maximizes the co-occurrence probability among the words that appear within a window, w, in a sentence (e.g tries to represent each word in a large text as a vector in a space of N dimensions which we will call features making similar words be close to each other)[26].

Breadth-first Sampling - It is a search strategy that searches only nodes that are immediate neighbors of the source. An example can be found in Figure 2.4.

Depth-first Sampling - It is a search strategy that consists of nodes sequentially sampled at increasing distances from the source node. An example can be found in Figure 2.4.

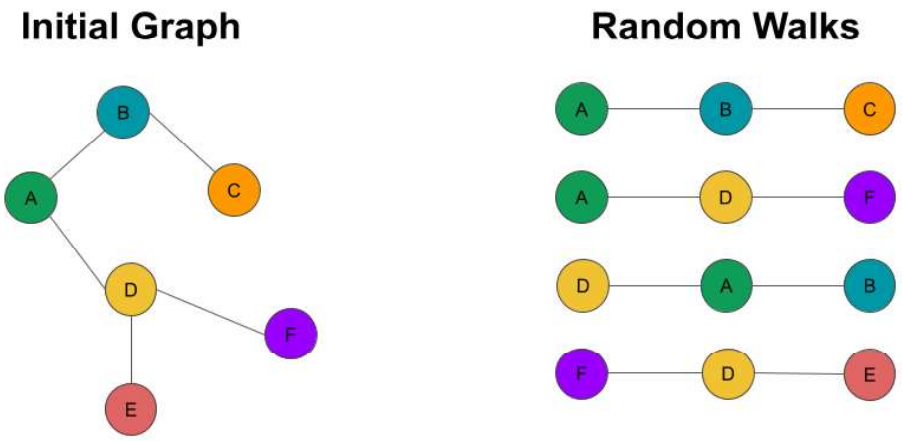


Figure 2.3: Random walks example.

2.3.2 DeepWalk

DeepWalk [32] is the first network embedding approach for learning latent representations of vertices in a network. It consists in two main components: a random walk generator and an update procedure. The workflow of DeepWalk is represented in Figure 2.5. Given a random walk sequence of length L,$\{v_1, v_2, \ldots, v_L\}$ DeepWalk learns the representation of vertex v_i estimating the likelihood of observing vertex v_i given all the previous vertices visited so far in the random walk. It is achieved by solving the optimization problem:

$$\min_{f} -logPr(\{v_{i-t,},...,v_{i+t,}\}\backslash v_i|\Phi(v_i)), \tag{2.1}$$

where $\{v_{i-t,},...,v_{i+t,}\} \setminus v_i$ are the context vertices of vertex v_i within t window size and $\Phi(v_i)$ represents the mapping function that associates each vertex v_i to it latent social representation. Making conditional independence assumption, Pr($\{v_{i-t,},...,v_{i+t,}\}$ is approximated as

$$Pr(\{v_{i-t,},...,v_{i+t,}\}\backslash v_i|f(v_i)) = \prod_{j=i-t,j\neq i}^{i+t} Pr(v_i|\Phi(v_i)). \tag{2.2}$$

Following the DeepWalk's learning architecture, vertices that share similar context vertices in random walk sequences must be represented closely in the new embedding space.

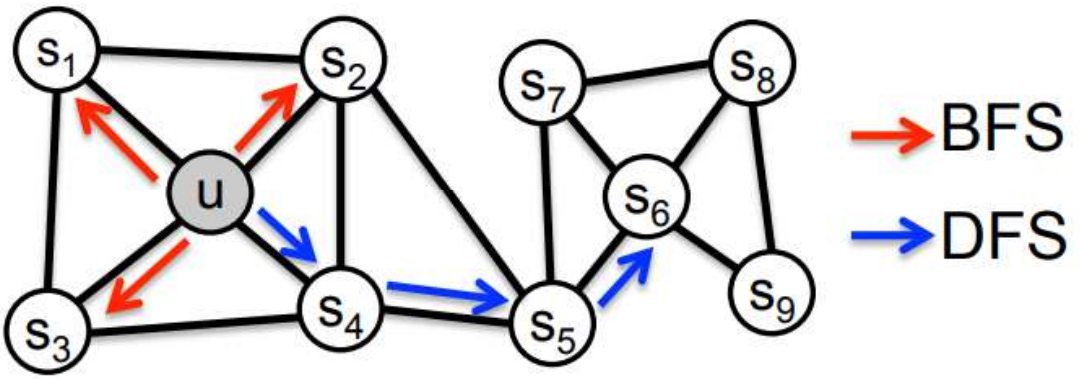


Figure 2.4: BFS and DFS search strategies from node u.

Considering the fact that context vertices in random walk sequences describe neighborhood structure, DeepWalk actually represents vertices sharing similar neighbors (direct or indirect) closely in the embedding space, so the second order and high-order proximity is preserved.

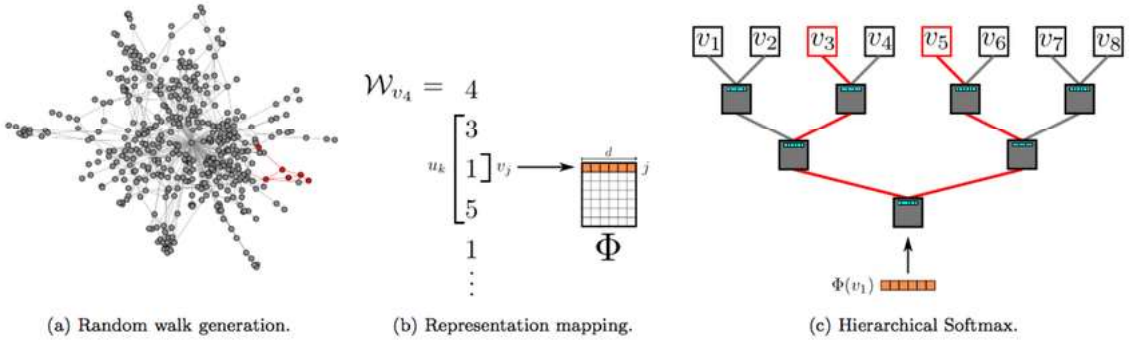


Figure 2.5: Deep Walk Overview [32].

2.3.3 Node2Vec

Besides DeepWalk, there is another algorithm that enable us to learn network embeddings: Node2vec [14]. Node2vec is a semi supervised algorithm for scalable feature learning. It uses a flexible neighborhood sampling strategy, i.e., biased random walk, which smoothly interpolates between two extreme sampling strategies, Breadth-first Sampling (BFS) and Depth-first Sampling (DFS).

The biased random walk can better preserve both the second-order and high-order prox-

imity. Following the skip-gram language model, given the set of neighbor vertices N (v_i) generated by biased random walk, node2vec learns the vertex representation $f(v_i)$ by optimizing the occurrence probability of neighbor vertices N (v_i) conditioned on the representation of vertex v_i,$f(v_i$ [14]):

$$\max_{f} \sum_{v_i \in V} logPr(N(v_i|f(v_i)). \tag{2.3}$$

2.3.4 LINE

Large-scale Information Network Embedding (LINE) [39], is an unsupervised algorithm that learns vertex representation by modeling, the first-order and second order proximity. Instead of exploiting random walks to capture network structure, learns vertex representations by explicitly modeling the first-order and second-order proximity.

To calculate the first-order proximity, for each vertex pair v_i and v_j we minimize using the following formula:

$$O_1 = \sum_{(i,v) \in E} w_{\mathrm{ij}} logp_1(v_i, v_j). \tag{2.4}$$

To calculate the second-order proximity, LINE minimizes the following formula:

$$O_2 = d(p - \sum_{(i,v) \in E} w_{\mathrm{ij}} logp_2(v_i|v_j). \tag{2.5}$$

After that a concatenation of the vector representations learned by LINE(1st order proximity) and LINE(2nd order proximity) into a longer vector is performed. After concatenation, the dimensions should be re-weighted to balance the two representations.

2.3.5 GraRep

GrapRep [7] algorithm follows the same approach of DeepWalk, it uses the skip-gram model to capture the high order proximity, i.e, vertex with sharing common k-step neighbors must be represented close to each other. It exploits the node co-occurrence at different scales by raising the adjacency matrix to different powers. Then SVD is applied to the powers of adjacency matrix to obtain a low-dimensional representation of nodes.

2.3.6 Summary

Network embeddings techniques are important to represent graphs and complex structures into lower dimensions, preserving all the information of vertex and edges. We present four different algorithms for embedding techniques DeepWalk, Node2Vec, LINE and GraRep. Each one has his advantages and disadvantages, three of them, Deepwalk, Node2Vec and GrapRep preserve more than two order representations whereas LINE only preserve first and second order proximity.

Chapter 3

Churn Predictive Modeling

In this chapter, we describe our churn prediction workflow and its steps. Figure 3.1 depicts this workflow where each of these blocks is explained in a different Section.

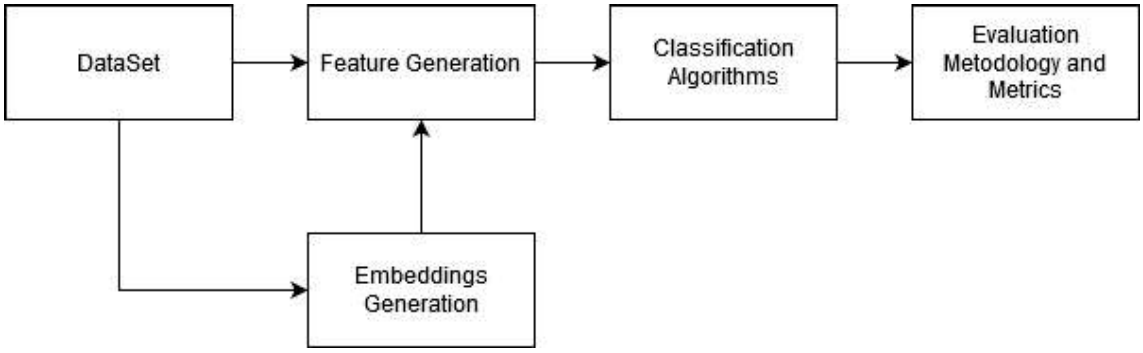


Figure 3.1: Churn prediction workflow.

We start by describing the data used in this work. Section 3.2 (Embeddings Generation) explains the generation of automatic features. Section 3.3 (Feature Generation) explains the generation of handcraft features while in Section 3.4 (Classification Algorithms) the algorithms used to generate the models are presented, and finally in Section 3.5 (Evaluation Methodology and Metrics) the evaluation methodology and metrics used to validate the models are detailed.

3.1 Dataset

In this work, we used an anonymized prepaid telecom dataset. This dataset is a mensal aggregation of call detail records (CDR) containing one year of information about customers communications, namely calls between customers and their duration and whether a customer has churned. The dataset contains 9.052.975 pairs and 1.784.083 unique nodes. A **pair** is a entry on our dataset which contains the features (**ID_A**, **ID_B**, **Month**, **Duration**, **Number_calls**). These pairs are a way to represent an adjacency list which represents a graph. A **node** is a unique customer who made calls to other customers. The Table 3.1 describes the initial dataset.

Name	Example	Description
ID_A	223c551ab	Id who made the call.
ID_B	b513c536	Id who receive the call.
Month	1	Month when the call was made.
Duration	2343	Duration in seconds of the calls.
Number_calls	26	Number of calls.
Churned_next_month	0 or 1	Indicate if the customer churned.
Churned_before	0 or 1	If the customer has churned before.

Table 3.1: Fields of one row of the dataset.

The described dataset was used to generate the embeddings described in the next Section 3.2 (Embeddings Generation).

3.2 Embeddings Generation

In this work we want to understand if a customer leaves what influence he has on his closest friends. To detect friends we assume that if a customer get in touch with another multiple times in a month it means that the probability of being a friend is high. For this it is necessary that when we want to represent a customer, we have access to information about the people with whom that customer relates to. So one of the ways suggested in this thesis is to represent that information in the form of embeddings. An embedding is an

array with multiple dimensions and we used Node2Vec algorithm to generate them. The algorithm was previously described on Chapter 2.3 (Network Embeddings). To generate the embeddings, first we aggregate all the pairs by year. After this process we have 1.784.083 nodes and 7.318.965 pairs. The number of calls between customers will be our weight between each node. The aggregation described allows us to reduce the number of pairs along with the complexity of the graph. Our graph is undirected which means that client **a** calling client **b** is the same that client **b** calling client **a**. This decision was made mainly to reduce complexity. After the graph creation, we generate the random walks using our graph. The parameters used to generate the random walks are described in Table 3.2. At the end of this process we have 2.497.716.200 random walks. The next

Parameter	Value	Description
n_random_walks_per_node	10	Number of random walks per node
random_walk_length	10	Length of each random walk
p	1.0	Probability, 1/p, of returning to source node
q	1.0	Probability, 1/q, for moving away from source node

Table 3.2: Random walk parameters.

step is the generation of embeddings with Node2vec. The parameters used in this process are described in Table 3.3. The final result of this process will be a matrix where m is

Parameter	Value	Description
window_size	10	Maximum distance between the current and predicted node within a graph
embedding_size	256	Dimensionality of the node vectors

Table 3.3: Node2vec parameters.

the unique node and n is the number of features generated. The matrix has a shape of (1784083, 256). In the next step, we calculate the similarity between pairs of embeddings using the cosine distance. This gives us the proximity of a node to each of the remaining nodes in our graph.

Having the matrix with the proximity between all nodes we then use it to create new features. This process is described in the next section.

3.3 Feature Generation

In addition to the features of the initial dataset, we will add the network embeddings to get a set of rich features this process is called feature enrichment. We created several features based on the number of calls, their duration and customer churn history. In this phase, we want to create data features that give some information about who are the closest "friends" of each client. Thus, first we filtered all the calls with less than 10 seconds assuming that friends tend to talk more than this threshold of seconds. Then all the rows with less than five calls each month were filtered out. This helps filtering the pairs of clients that have sporadic communications, which we assume is different from what friends usually do. After this process we get 1784083 unique nodes and 7318965 pairs. Features were then created on this filtered dataset. We segment the features by: Churners vs Non-Churners and Time.

Non-churner features - We generate these features because we want to know what are the closest friends of each node. To do that, we use duration and the number of calls. We assume that the more number of calls and his duration the higher probability of friendship. Table A.1 describes features related to duration, number of calls and distinct connections between non churner friends.

Churner features - These features allow us to know the friends that churned. To create this features we filter all the customers that churned, after that we calculate the duration and the number of calls for each node. We are expecting that if we have a lot of friends connected that churned the probability of this node churning is high. The Table A.2 describes features related to churner friends.

Churner features by time - In these features we want to calculate the friends that churned recently. We are assuming that if a friend churned recently, the customer is more likely to churn. To accomplish that we generate features that takes the time when the friend churn into consideration. We are considering one and two months. The Table A.3 describes features related to churner friends aggregated per month.

The features generated are essentially counts and top of customers, based on the number of calls, duration of calls, percentage of churners and time. After this process we have 119 new features in our dataset, which will be our baseline for our prediction models.

Finally we add our embedding features that we create in the previous chapter. Our embedding features are described below.

Churner features with embeddings similarity - This features allow us to know the friends that churned with the embeddings similarity to other friends. The Table B.1 describes these features.

Churner features with embeddings similarity by time - In this features we want to calculate the friends that churned recently with the similarity to other friends. The Table B.2 describes these features.

After this process we have more 75 features to add to our model.

3.4 Classification Algorithms

After we generated the features for our work we perform a normalization using z-score [17] because some classification algorithms are sensitive to scale.

We tried two different supervised machine learning algorithms to create our models:

Logistic Regression: is a statistical model [17] that can be used to determine if an independent variable has an effect on a binary dependent variable. This means that there are only two potential outcomes given an input. For example, it may be used to determine if an email is spam, or not, using the rate of misspelled words, a common sign of spam.

XGBoostTree: Extreme Gradient Boosting [10, 9] is an implementation of gradient boosted decision trees designed for speed and performance.

We select this one because Logistic Regression is a simple and interpretable model which can achieve good results, but XGboostTree is a more advance technique that tends to produce state-of-the-art results in some machine learning challenges with tabular data. To train our models we had to choose the best hyper parameters [11]. There are multiple approaches for hyperparameter tuning such as: Random search [5], Grid Search [5], Bayesian optimization [37],Gradient-based optimization [25], etc. We choose Grid Search approach.

3.5 Evaluation Methodology and Metrics

To know the performance of our classification algorithms we need to evaluate the models. We perform a 1 month split in the dataset: 80% for training and 20% for testing. For training we use 141.277 data entries, where 6351 are churners clients in the next month. Which means that 4.5% of the dataset are churners. In this thesis cross validation was not used. We use approximately ≈8000 samples for testing.

After the training process the churn prediction system outputs a list of customers(non churners in the current month) that will churn in the next month. To evaluate the model we use the metrics below:

Lift curve: The lift curve describes how well the model behave. It is the ratio of customers that churn on a certain point divided by the ratio of churners on the whole dataset [17]:

$$lift = \frac{PredictedRate}{Averagerate} \tag{3.1}$$

Precision at K[20]: Precision at k is the proportion of churners in the top-k elements with higher probability returned by the model. As an example, if our precision at 10 is 80%, this means that 8 out of 10 were correctly predicted as churners[17].

$$Precision@K = \frac{\#\ of\ churners\ @k}{k} \tag{3.2}$$

Recall at K[20]: Recall at k is the proportion of churners found in the top-k from all the existing churners. For example, a recall at 10 of 40% means that is used 40% of the total number of appear in the top-k results[17].

$$Recall@K = \frac{\#of\ churners\ @k}{total\#churners} \tag{3.3}$$

Area Under ROC Curve[20]: AUC - ROC curve is a performance measurement for the classification problem at various thresholds settings. ROC is a probability curve and AUC represents degree or measure of separability between classes[17]. Higher the AUC,

better the model is at distinguishing between churners and non churners. AUC is given by the followed function:

$$AUC = \frac{\sum_n \in TPRank_{\mathrm{n}} - \frac{P \times (P+1)}{2}}{P \times N} \tag{3.4}$$

where P is the number of true churners, and N the number of true non-churners sorting the churner likelihood in descending order. For $TP\ Rank_{\mathrm{n}}$ we, sort the churner likelihood in descending order and assign the highest likelihood customer with the rank n, the second highest likelihood customer with the rank n - 1 and so on.

These metrics are the most common used in churn prediction. The next chapter describes the results and the discussion about the metrics explained in this chapter.

Chapter 4

Results

In this chapter, we will go into detail about the conducted experiments and the results achieved in each experiment.

The Section 4.1 (Experiments) describes the experiments made in this work. The Section 4.2 (Feature Analysis) describes the feature analysis performed before applying the classification churn models. The Section 4.3 (Model Results) presents the predictive results using baseline features and network embeddings. Finally, in Section 4.4 (Discussion) there is a discussion about the results presented along with the findings found in this work.

4.1 Experiments

Since this thesis focuses on understanding whether embeddings can help detect churn, we decided to create two experiments in order to detect their impact. The first creates a baseline model, where embeddings features are not included in order to realize what results are possible to achieve. In a second experience, we added the embeddings and measure the impact this has on the results. In the baseline model, features are mostly counts between number of calls, duration of calls, and percentage of churners in the top friends (with most communications) described in Chapter 3.3 (Feature Generation). To select the best features for our model performed create a feature selection analysis. In this process we want to know the correlation between the features. We found that multiple features are correlated above 90%. We discarded the most correlated features and remained 42

features for our experiments. The next step is training, and for that we split the dataset into 80% of the data for training and 20% for testing. We applied the algorithms Logistic Regression and XGBoost, described in Chapter 3.4 (Classification Algorithms). Then we evaluated our classification models using Precision at K, Recall at K, Lift curves and AUC. All these metrics are explained in Chapter 3.5 (Evaluation Methodology and Metrics)

We then created a second experiment with the baseline features described before and added more 14 features. These 14 features are created from embeddings similarity. At the end of this stage we had 56 features to train our classification models. The training and evaluation processes were the same described in the first experiment. The results of these two experiments are described in Chapter 4.3 (Model Results).

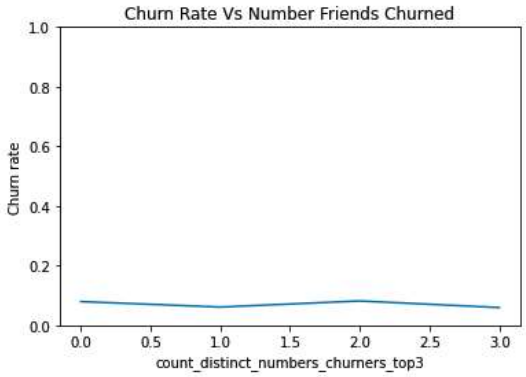
4.2 Feature Analysis

Before presenting the results of our models we perform a feature analysis. The analysis compares the features created against the churn rate. We split these analysis in **Baseline Features** and **Network Embedding Features** where we compare their performance.

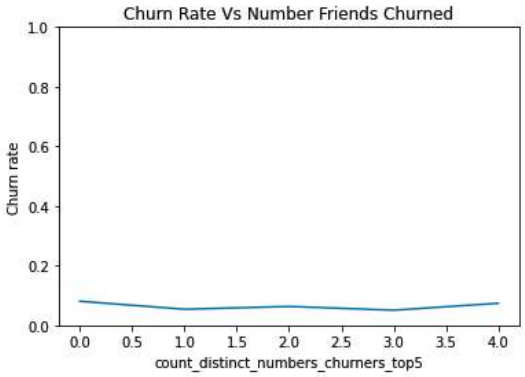
For baseline feature analysis we can see in Figure 4.1 the percentage of churn rate with respect to the top 3, 5, 10 friends and also to all customers contacted in one year. As we can see there is not a trend between the number of friends that churned and churn rate, with a slightly exception in Figure 4.1 (c) after 5 friends that churned. In this exception we can detect more churners as the number of their friends who churned increases.

The Figure 4.2 represent the percentage of churn rate with respect to top 3, 5, 10 and also all customers contacted when friends churned in the last month. We can see in all the figures that there is a growing trend of churn rate as the number of friends churned in the last month. These Figures give a clear indication that friends who churned have an impact in customers decision to change to a different telecommunication operator. These churn rates can grow to percentages higher than 50%, such as in Figure 4.2 (c).

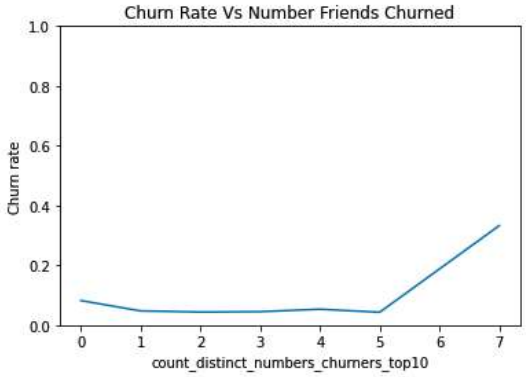
The Figures 4.3 represents the percentage of churn rate with respect to top 3, 5, 10 and all customers when friends churned in the last two months. Figures 4.3 (a), 4.3 (b) and 4.3 (c) show also a growing trend of the churn rate, correlated with the number of friends churning in the closer two months.



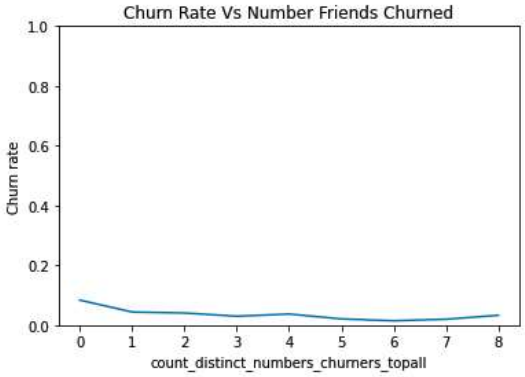
(a) Churn rate vs top 3 churned friends



(b) Churn rate vs top 5 churned friends



(c) Churn rate vs top 10 churned friends

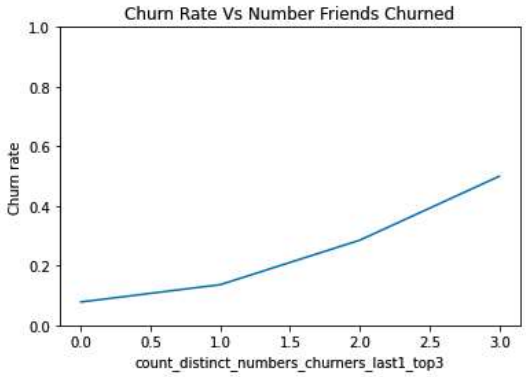


(d) Churn rate vs top all churned friends

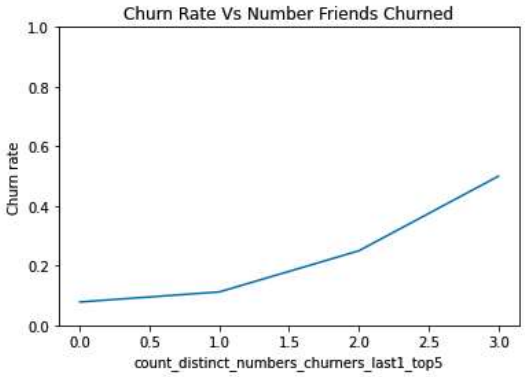
Figure 4.1: Churn Rate vs top 3, 5, 10 and all churned friends with baseline features.

Comparing the plots depicted in Figures 4.4 with baseline features against those in Figure 4.1 embedding features we can see clearly that with network embeddings the churn rate increases. This indicates that with network embeddings we can discover more churners than with baseline features.

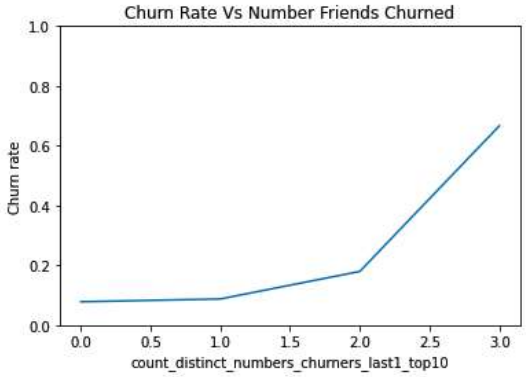
Regarding Figures 4.2 with baseline features and 4.5 with embedding features represents the percentage of churn rate with respect to top 3, 5 and 10 customers when friends churn in the last month. The Figure 4.2 (a) shows a increasing trend as churn rate increases the number of customers churned increase, although 4.5 (a) shows an increasing trend at the beginning and after 1 churned customer the churn rate starts decreasing. After two churner friends the churn rate increases exponentially. Regarding the top 5 and top 10 plots in Figure 4.5 (b) and 4.5 (c) using embeddings we are able to detect more churner friends.



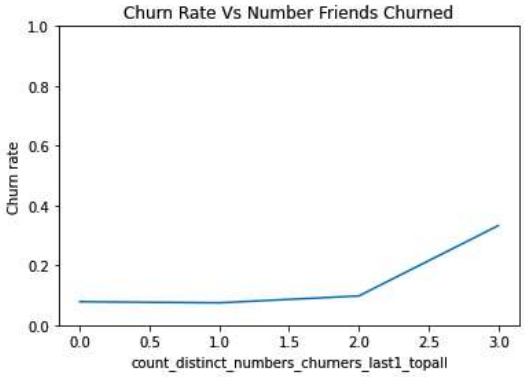
(a) Churn rate vs last month top 3 churned friends



(b) Churn rate vs last month top 5 churned friends



(c) Churn rate vs last month top 10 churned friends

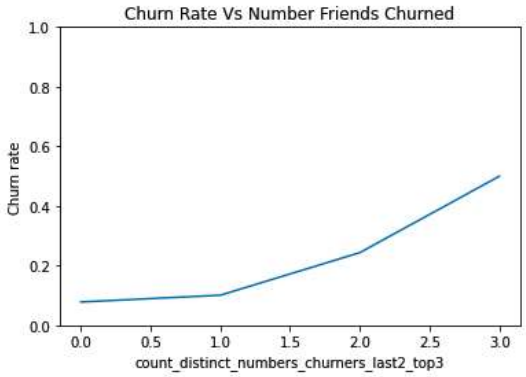


(d) Churn rate vs last month top all churned friends

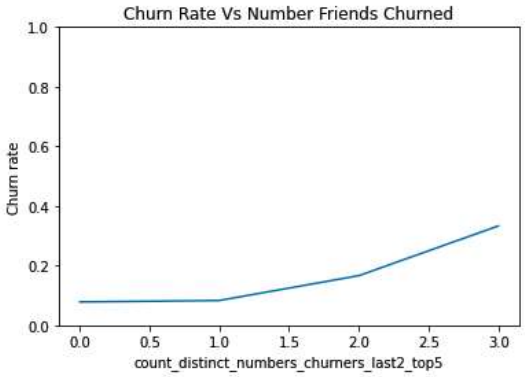
Figure 4.2: Churn rate vs last month top 3, 5, 10 and all churned friends when churned in the last month with baseline features.

Finally comparing Figures 4.3 and 4.6 represents the percentage of churn rate with respect to top 3, 5 and 10 customers when friends churn in the last two months. Comparing the top 3 we detect the more number of churners, with embeddings than with baseline. Comparing top 5 we detect more churners with network embeddings, comparing the top 10 we are able to detect more churners with network embeddings.

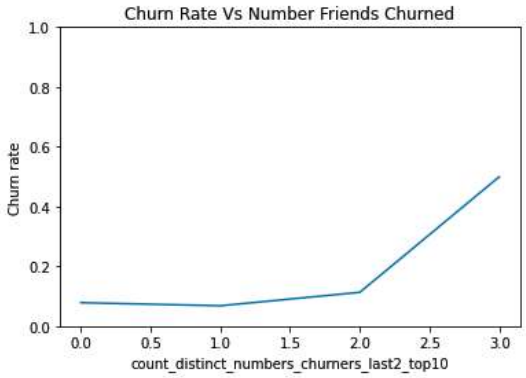
As we can saw in the feature analysis the baseline features detect a smaller churn rate comparing with network embeddings features. This give us the indication that network embeddings will likely outperform typical social network analysis. The next section describes the results of our models.



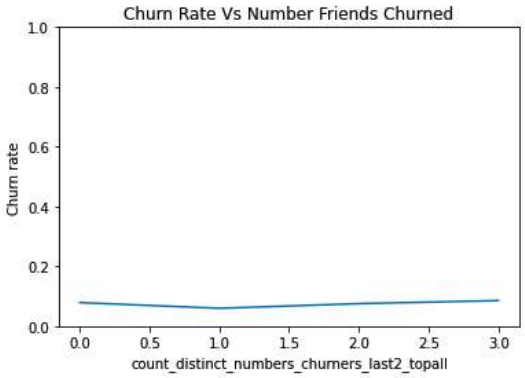
(a) Churn rate vs last two months top 3 churned friends



(b) Churn rate vs last two months top 5 churned friends



(c) Churn rate vs last two months top 10 churned friends

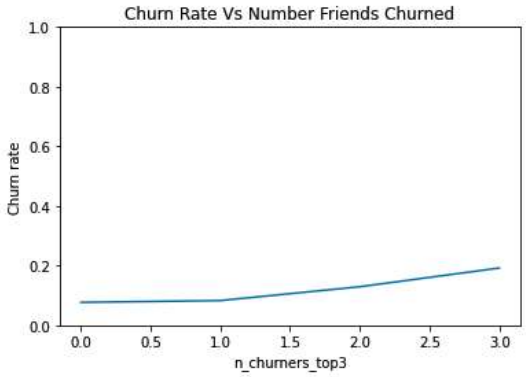


(d) Churn rate vs last two months top all churned friends

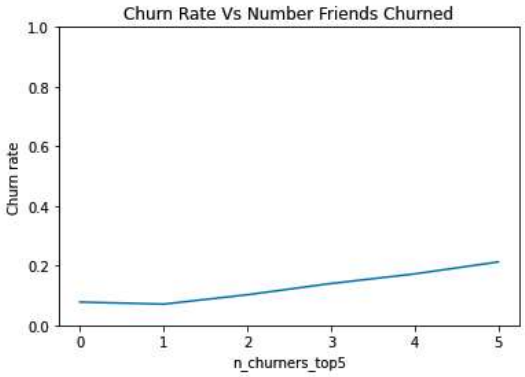
Figure 4.3: Churn rate vs last month top 3, 5, 10 and all churned friends in the last two months with baseline features.

4.3 Model Results

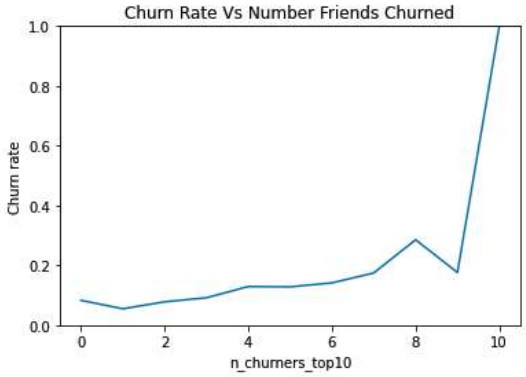
This section presents the results of our models, we present the results using the metrics **precision at k**, **recall at k**, **AUC** and **lift curves** that are detailed in Chapter 3.5 (Evaluation Methodology and Metrics). Table 4.1 presents the evaluation results for our baseline models. We achieved 92% and 55% on precision predicting churn for our top 50 and 100 customers with higher propensity score using XGBoost. This value tend to decrease when we add more samples. Using Logistic Regression algorithm, the behavior is slightly different, it starts lower and tend to increase. Table 4.2 presents the evaluation results for our models with network embeddings. We achieved 96% and 64% on precision predicting churn in our top 50 and 100 customers using XGBoost algorithm. Comparing



(a) Churn rate vs top 3 churned friends



(b) Churn rate vs top 5 churned friends



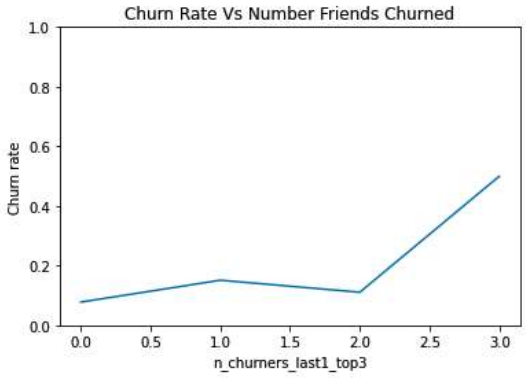
(c) Churn rate vs top 10 churned friends

Figure 4.4: Churn Rate vs top 3, 5 and 10 churned friends with embedding features.

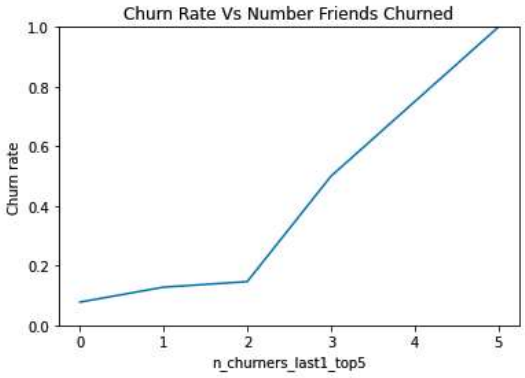
these results with baseline features, we can see that with network embeddings we are able to achieve 8% more on precision at top 50 customers and better results in most cut-offs of precision. This show that with network embeddings features we achieve better results.

The Figure 4.7 represents our AUC results for logistic regression and XGBboost algorithms. Figure 4.7 (a) represents the area under the curve for baseline features with 77.57% for Logistic Regression and 78.57% for XGBoost. The Figure 4.7 (b) depicts the area under the curve for embedding features. We achieved 78.223% for Logistic Regression and 80.61% for XGBoost. We can conclude that using XGBoost we achieve a better performance on predicting churn. Additionally, both results achieve better results with embeddings features. XGBoost gets an AUC improvement of 2% using embeddings.

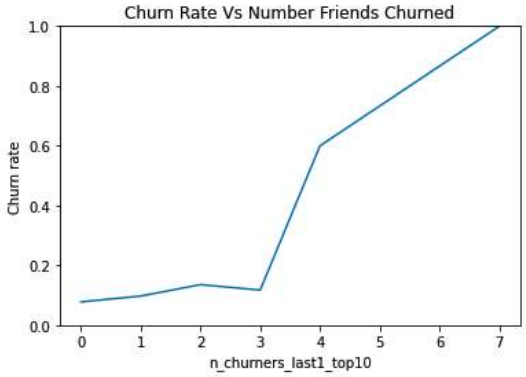
Figure 4.8 represents our lift curves for baseline and embedding features. Again XGBoost achieved the best performance.



(a) Churn rate vs last month top 3 churned friends



(b) Churn rate vs last month top 5 churned friends



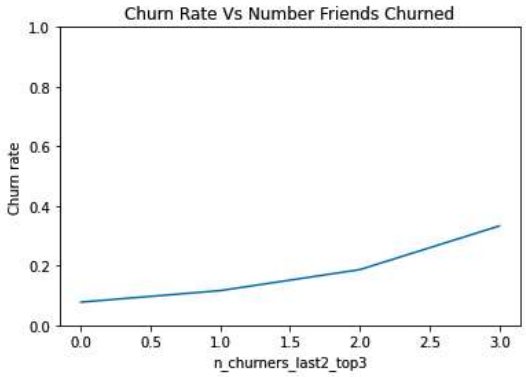
(c) Churn rate vs last month top 10 churned friends

Figure 4.5: Churn Rate vs top 3, 5 and 10 last month churned friends with embedding features.

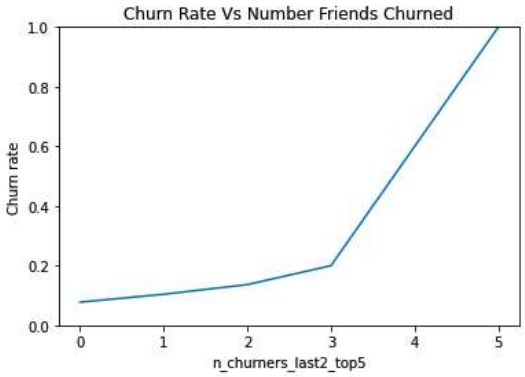
After analyzing the algorithms we went to see the best model, which features are most important for the classification process. The Figure 4.9 presents the feature importance for our baseline features . As we can see the features with more impact for our models is the number of calls in top 3 and top 5. Figure 4.10 presents the feature importance for our baseline features as we can see the features with more impact for our models is the number of calls in top 3 and top 5.

4.4 Discussion

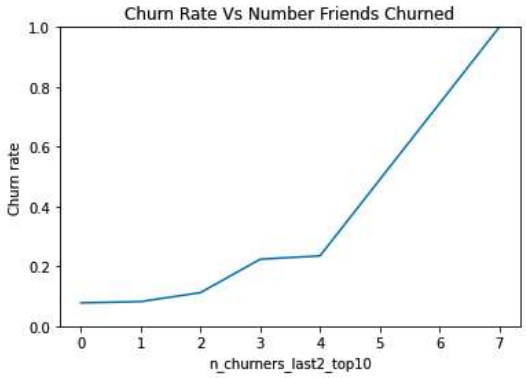
Our results show, that we can discover more churners using network embeddings. This is visible on the graphics in Chapter 4.2 (Feature Analysis) where we observed that with



(a) Churn rate vs last two months top 3 churned friends



(b) Churn rate vs last two months top 5 churned friends



(c) Churn rate vs last two months top 10 churned friends

Figure 4.6: Churn Rate vs top 3, 5 and 10 last two months churned friends with embedding features.

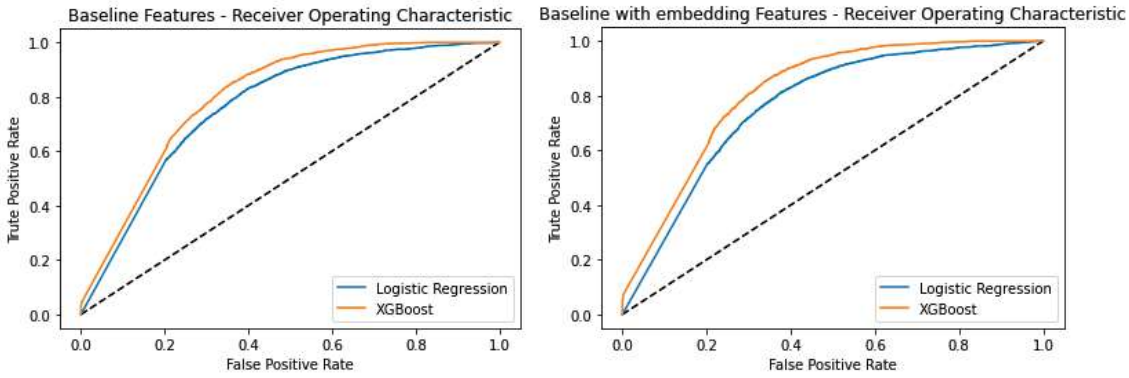
network embeddings the churn rate increases as number of friends. Regarding baseline features vs embedding features, we achieved better results with embedding features, XGBoost algorithm outperform logistic regression both using baseline features and network embedding features. We believe that this is due to the algorithm nature, Logistic Regression is a algorithm more simpler than XGBoost. We also noted that the better P@K and R@K is observed especially in the beginning. Regarding AUC ROC curves we achieve better performance with network embeddings where XGBoost always outperform logistic regression. Finally our Lift Curves demonstrate that with a sample of 0.1% we are able to detect 0.12% of churners. We can conclude that node embedding techniques can help on churn prediction this results demonstrate a great potential of these techniques.

	Logistic Reg		**Xgboost**	
Sample/K	**P@K**	**R@k**	**P@K**	**R@k**
50	0.0550	0.7050	**0.9200**	**0.912**
100	0.0530	0.6111	**0.5500**	**0.5445**
150	0.0500	0.4705	0.3866	0.3647
200	0.0650	0.3703	0.3250	0.2901
300	0.0666	0.3414	0.2433	0.2457
400	0.0700	0.2743	0.2000	0.2122
500	0.0780	0.2466	0.1740	0.1875
1000	0.0870	0.3392	0.1210	0.1568
5000	0.0752	0.6362	0.0912	0.1180
10000	0.0734	0.5571	0.0815	0.1091

Table 4.1: Evaluation results: Baseline features.

	Logistic Reg		**Xgboost**	
Sample/K	**P@K**	**R@k**	**P@K**	**R@k**
50	0.0600	0.7153	**0.9600**	**0.9550**
100	0.0600	0.6666	**0.6400**	**0.5614**
150	0.0600	0.5000	0.4466	0.3701
200	0.0650	0.4193	0.3550	0.2817
300	0.0666	0.3921	0.2566	0.2340
400	0.0700	0.3544	0.2150	0.2072
500	0.0780	0.3305	0.1880	0.1846
1000	0.0870	0.4243	0.1420	0.2181
5000	0.0752	0.6471	0.0862	0.3983
10000	0.0734	0.5581	0.0785	0.4204

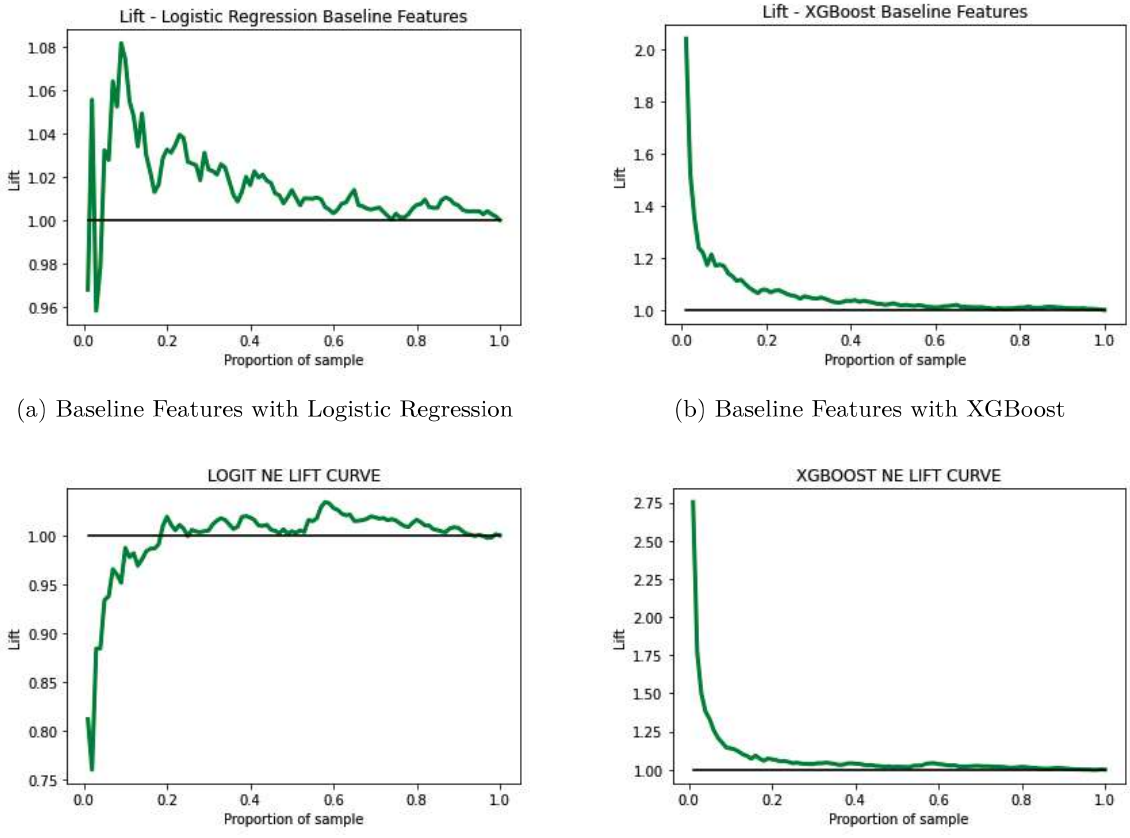
Table 4.2: Evaluation results with baseline and embedding features.



(a) AUC with Baseline Features

(b) AUC with baseline and embedding Features

Figure 4.7: AUC Baseline and Network Embedding features.



(a) Baseline Features with Logistic Regression

(b) Baseline Features with XGBoost

(c) Network Embedding Features with Logistic Regression

(d) Network Embedding Features with XGBoost

Figure 4.8: Lift Curves Baseline and Network Embedding features.

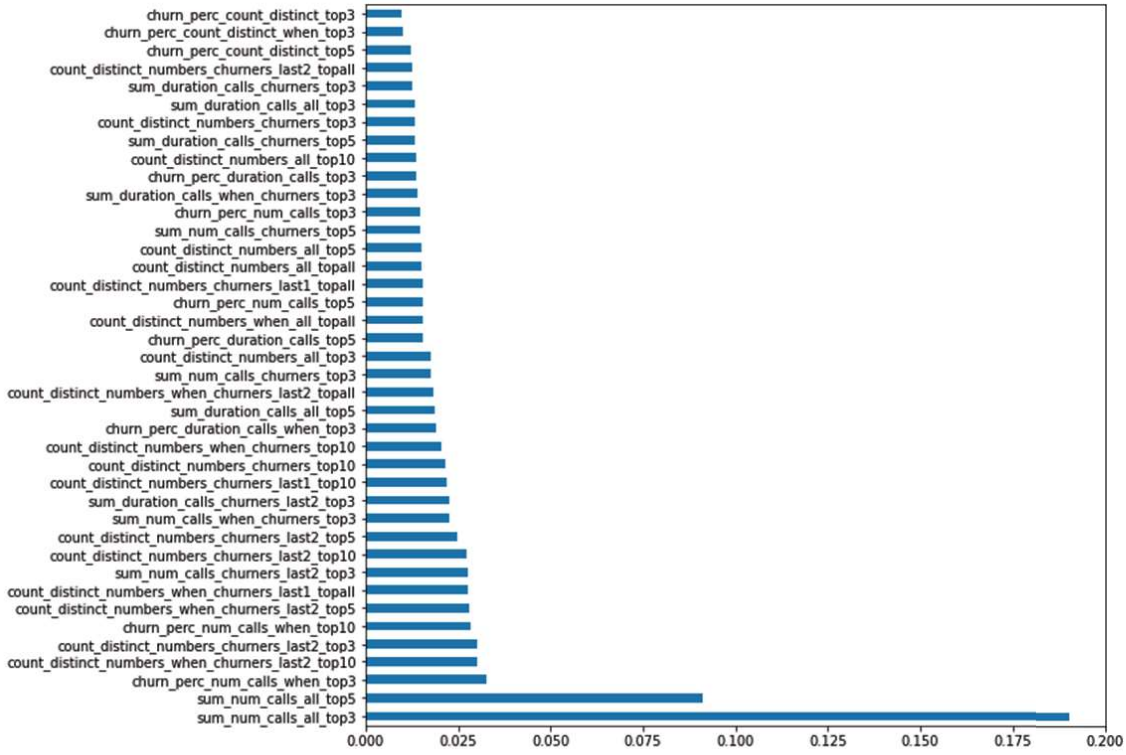


Figure 4.9: Feature Importance XGBoost Baseline Features.

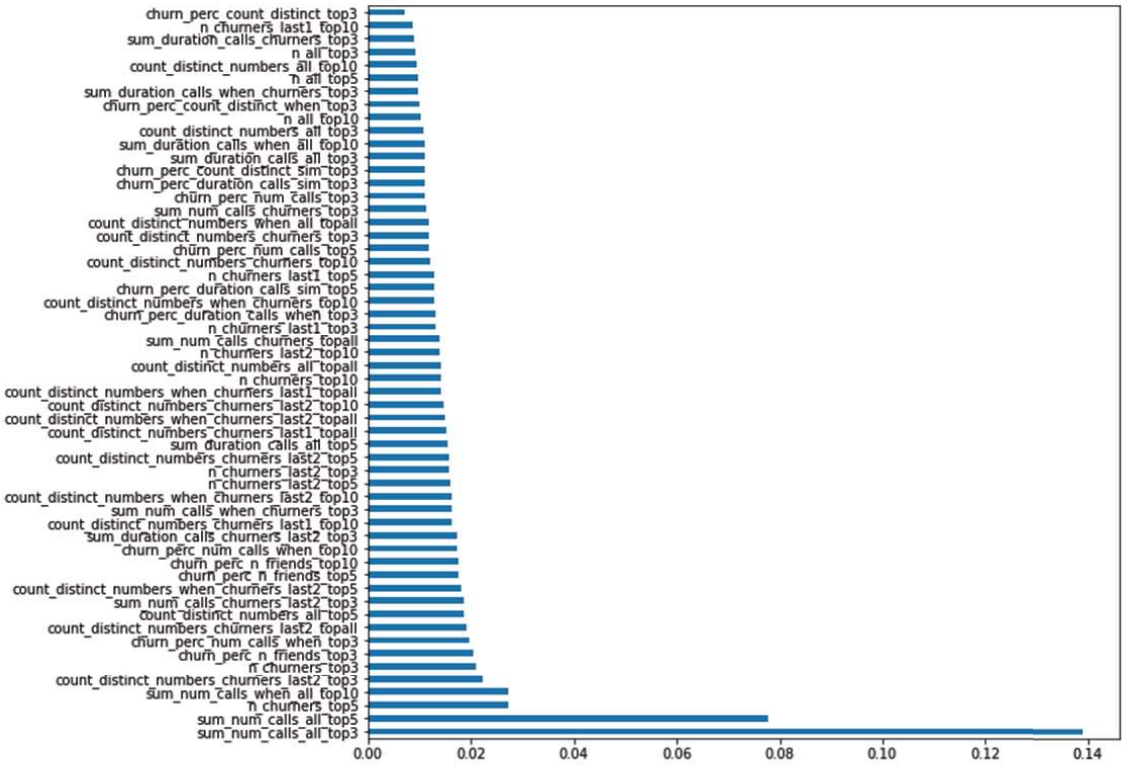


Figure 4.10: Feature Importance XGBoost Network Embeddings

Chapter 5

Conclusion and Future Work

With the exponential increase in the use of the Internet, customers began to understand better what services they hire and what companies offer. In this context, companies began to worry about the turnover between operators. In this thesis we intend to predict the turnover between customers using only the connections between customers.

We developed a baseline model from the dataset that contained two features: number of calls and duration of calls. From these two features, manual features were created. This dataset was used and the XGBoost algorithm. With this baseline model we were able to achieve an precision of 92% in a sample of 50 customers.

In this project we wanted to understand if using the concept of friendship we could improve the performance of the churn prediction system. For this purpose, in addition to the features created manually, network embeddings features were added to the system. Using the same classification algorithm and adding the embeddings features to the baseline, we were able to increase performance by 2%.

Or feature analysis demonstrates that after 2 months the probability of churn goes down. The features that contribute more for our model is the number of calls, and the number of churners. Leaving us to the conclusion that the number of calls and churner friends has a higher impact on customer leaving the company. As far as we know this kind of work was never done before.

One of the interesting notes that we discovered during the project is influence of friends is goes down over time. The probability of a friend leaving the operator after 1 month

is greater than after 12 months. As with future work, since time may have an influence in churn, it would be interesting to explore the concept of dynamic network embeddings, since they take into account the evolution of the network of friends over time.

Appendix A

Handcraft Features

Name	Description
count_numbers_top3	Number of connections within the top 3 more connected (max 3)
count_numbers_top5	Number of connections within the top 5 more connected (max 5)
count_numbers_top10	Number of connections within the top 10 more connected (max 10)
count_numbers_topall	Number of connections within all more connected
sum_duration_calls_top3	Sum duration of the calls that are in top 3
sum_duration_calls_top5	Sum duration of the calls that are in top 5
sum_duration_calls_top10	Sum duration of the calls that are in top 10
sum_duration_calls_topall	Sum duration of the calls that are in top 3
sum_num_calls_top3	Sum number of calls select top 3 with more calls
sum_num_calls_top5	Sum number of calls select top 5 with more calls
sum_num_calls_top10	Sum number of calls select top 10 with more calls
sum_num_calls_topall	Sum number of calls select the top all with more calls

Table A.1: Handcraft features with respect to friends.

Name	Description
count_numbers_churners_top3	Number of churner in the top 3 numbers most connected
count_numbers_churners_top5	Number of churner in the top 5 numbers most connected
count_numbers_churners_top10	Number of churner in the top 10 numbers most connected
count_numbers_churners_topall	Number of churner connected
count_numbers_when_churners_top3	Count top 3 churners when churners
count_numbers_when_churners_top5	Count top 5 churners when churners
count_numbers_when_churners_top10	Count top 10 churners when churners
count_numbers_when_churners_topall	Count top all churners when churners
sum_duration_calls_churners_top3	Sum duration calls top 3 churners
sum_duration_calls_churners_top5	Sum duration calls top 5 churners
sum_duration_calls_churners_top10	Sum duration calls top 10 churners
sum_duration_calls_churners_topall	Sum duration calls top all churners
sum_duration_calls_when_churners_top3	Sum duration calls top 3 churners when churners
sum_duration_calls_when_churners_top5	Sum duration calls top 5 churners when churners
sum_duration_calls_when_churners_top10	Sum duration calls top 10 churners when churners
sum_duration_calls_when_churners_topall	Sum duration calls top all churners when churners
sum_num_calls_churners_top3	Sum number calls top 3 churners
sum_num_calls_churners_top5	Sum number calls top 5 churners
sum_num_calls_churners_top10	Sum number calls top 10 churners
sum_num_calls_churners_topall	Sum number calls top all churners
sum_num_calls_when_churners_top3	Sum number calls top 3 churners when churners
sum_num_calls_when_churners_top5	Sum number calls top 5 churners when churners
sum_num_calls_when_churners_top10	Sum number calls top 10 churners when churners
sum_num_calls_when_churners_topall	Sum number calls top all churners when churners
churn_perc_num_calls_top3	Churn % number of calls top 3
churn_perc_num_calls_top5	Churn % number of calls top 5
churn_perc_num_calls_top10	Churn % number of calls top 10
churn_perc_num_calls_topall	Churn % number of calls top all
churn_perc_duration_calls_top3	Churn % duration of calls top 3
churn_perc_duration_calls_top5	Churn % duration of calls top 5
churn_perc_duration_calls_top10	Churn % duration of calls top 10
churn_perc_duration_calls_topall	Churn % duration of calls top all
churn_perc_count_top3	Churn % count top 3
churn_perc_count_top5	Churn % count top 5
churn_perc_count_top10	Churn % count top 10
churn_perc_count_topall	Churn % count top all
count_numbers_when_top3	Count the top numbers when last month in top 3
count_numbers_when_top5	Count the top numbers when last month in top 5
count_numbers_when_top10	Count the top numbers when last month in top 10
count_numbers_when_topall	Count the top numbers when last month all
sum_duration_calls_when_top3	Sum duration between the calls last month on top3
sum_duration_calls_when_top5	Sum duration between the calls last month on top5
sum_duration_calls_when_top10	Sum duration between the calls last month on top10
sum_duration_calls_when_topall	Sum duration between the calls last month on all

Name	Description
count_numbers_churners_last1_top3	Count top 3 churners last month
count_numbers_churners_last1_top5	Count top 5 churners last month
count_numbers_churners_last1_top10	Count top 10 churners last month
count_numbers_churners_last1_topall	Count all churners last month
count_numbers_churners_last2_top3	Count top 3 churners last two months
count_numbers_churners_last2_top5	Count top 5 churners last two months
count_numbers_churners_last2_top10	Count top 10 churners last two months
count_numbers_churners_last2_topall	Count top all churners last two months
count_numbers_when_churners_last1_top3	Count top 3 churners last month when churners
count_numbers_when_churners_last1_top5	Count top 5 churners last month when churners
count_numbers_when_churners_last1_top10	Count top 10 churners last month when churners
count_numbers_when_churners_last1_topall	Count top all churners last month when churners
count_numbers_when_churners_last2_top3	Count top 3 churners last two month when churners
count_numbers_when_churners_last2_top5	Count top 5 churners last two month when churners
count_numbers_when_churners_last2_top10	Count top 10 churners last two month when churners
count_numbers_when_churners_last2_topall	Count top all churners last two month when churners
sum_duration_calls_churners_last1_top3	Sum churners duration calls last month top 3
sum_duration_calls_churners_last1_top5	Sum churners duration calls last month top 5
sum_duration_calls_churners_last1_top10	Sum churners duration calls last month top 10
sum_duration_calls_churners_last1_topall	Sum churners duration calls last month top all
sum_duration_calls_churners_last2_top3	Sum churners duration calls last two months top 3
sum_duration_calls_churners_last2_top5	Sum churners duration calls last two months top 5
sum_duration_calls_churners_last2_top10	Sum churners duration calls last two months top 10
sum_duration_calls_churners_last2_topall	Sum churners duration calls last two months top all
sum_duration_calls_when_churners_last1_top3	Sum duration calls top 3 churners last month when churners
sum_duration_calls_when_churners_last1_top5	Sum duration calls top 5 churners last month when churners
sum_duration_calls_when_churners_last1_top10	Sum duration calls top 10 churners last month when churners
sum_duration_calls_when_churners_last1_topall	Sum duration calls top all churners last month when churners
sum_duration_calls_when_churners_last2_top3	Sum duration calls top 3 churners last two months when churners
sum_duration_calls_when_churners_last2_top5	Sum duration calls top 5 churners last two months when churners
sum_duration_calls_when_churners_last2_top10	Sum duration calls top 10 churners last two months when churners
sum_duration_calls_when_churners_last2_topall	Sum duration calls top all churners last two months when churners
sum_num_calls_churners_last1_top3	Sum number of calls top 3 churners last month
sum_num_calls_churners_last1_top5	Sum number of calls top 5 churners last month
sum_num_calls_churners_last1_top10	Sum number of calls top 10 churners last month
sum_num_calls_churners_last1_topall	Sum number of calls top all churners last month
sum_num_calls_churners_last2_top3	Sum number of calls top 3 churners last two months
sum_num_calls_churners_last2_top5	Sum number of calls top 5 churners last two months
sum_num_calls_churners_last2_top10	Sum number of calls top 10 churners last two months
sum_num_calls_churners_last2_topall	Sum number of calls top all churners last two months
sum_num_calls_when_churners_last1_top3	Sum number of calls top 3 churners last month when churners
sum_num_calls_when_churners_last1_top5	Sum number of calls top 5 churners last month when churners
sum_num_calls_when_churners_last1_top10	Sum number of calls top 10 churners last month when churners
sum_num_calls_when_churners_last1_topall	Sum number of calls top all churners last month when churners
sum_num_calls_when_churners_last2_top3	Sum number of calls top 3 churners last two months when churners
sum_num_calls_when_churners_last2_top5	Sum number of calls top 5 churners last two months when churners
sum_num_calls_when_churners_last2_top10	Sum number of calls top 10 churners last two months when churners
sum_num_calls_when_churners_last2_topall	Sum number of calls top all churners last two months when churners

Table A.3: Handcraft features with respect to churner friends aggregated per month.

Appendix B

Embedding Features

Name	Description
count_numbers_sim_all_top3	Count pairs similarity all top 3
count_numbers_sim_all_top5	Count pairs similarity all top 5
count_numbers_sim_all_top10	Count pairs similarity all top 10
count_numbers_sim_all_topall	Count pairs similarity all top all
count_numbers_sim_churners_top3	Count pairs similarity churners top 3
count_numbers_sim_churners_top5	Count pairs similarity churners top 5
count_numbers_sim_churners_top10	Count pairs similarity churners top 10
count_numbers_sim_churners_topall	Count pairs similarity churners top all
sum_duration_calls_sim_all_top3	Sum duration of calls similarity all top 3
sum_duration_calls_sim_all_top5	Sum duration of calls similarity all top 5
sum_duration_calls_sim_all_top10	Sum duration of calls similarity all top 10
sum_duration_calls_sim_all_topall	Sum duration of calls similarity all top all
sum_duration_calls_sim_churners_top3	Sum duration of calls similarity churners top 3
sum_duration_calls_sim_churners_top5	Sum duration of calls similarity churners top 5
sum_duration_calls_sim_churners_top10	Sum duration of calls similarity churners top 10
sum_duration_calls_sim_churners_topall	Sum duration of calls similarity churners top all
sum_num_calls_sim_all_top3	Sum number of calls similarity all top 3
sum_num_calls_sim_all_top5	Sum number of calls similarity all top 5
sum_num_calls_sim_all_top10	Sum number of calls similarity all top 10
sum_num_calls_sim_all_topall	Sum number of calls similarity all top all
sum_num_calls_sim_churners_top3	Sum number of calls similarity churners top 3
sum_num_calls_sim_churners_top5	Sum number of calls similarity churners top 5
sum_num_calls_sim_churners_top10	Sum number of calls similarity churners top 10
sum_num_calls_sim_churners_topall	Sum number of calls similarity churners top all
churn_perc_num_calls_sim_top3	Churn percentage number of calls similarity number of top 3
churn_perc_num_calls_sim_top5	Churn percentage number of calls similarity number of top 5
churn_perc_num_calls_sim_top10	Churn percentage number of calls similarity number of top 10
churn_perc_num_calls_sim_topall	Churn percentage number of calls similarity number of top all
churn_perc_duration_calls_sim_top3	Churn percentage duration of calls similarity number of top 3
churn_perc_duration_calls_sim_top5	Churn percentage duration of calls similarity number of top 5
churn_perc_duration_calls_sim_top10	Churn percentage duration of calls similarity number of top 10
churn_perc_duration_calls_sim_topall	Churn percentage duration of calls similarity number of top all
churn_perc_count_sim_top3	Churn percentage count similarity number of top 3
churn_perc_count_sim_top5	Churn percentage count similarity number of top 5
churn_perc_count_sim_top10	Churn percentage count similarity number of top 10
churn_perc_count_sim_topall	Churn percentage count similarity number of top all
n_all_top3	Number all top 3
n_all_top5	Number all top 5
n_all_top10	Number all top 10
n_churners_top3	Number churners top 3
n_churners_top5	Number churners top 5
n_churners_top10	Number churners top 10
churn_perc_n_friends_top3	Churn percentage number of top 3 friends
churn_perc_n_friends_top5	Churn percentage number of top 5 friends
churn_perc_n_friends_top10	Churn percentage number of top 10 friends

Name	Description
count_numbers_sim_churners_last1_top3	Count pairs similarity churners last month top 3
count_numbers_sim_churners_last1_top5	Count pairs similarity churners last month top 5
count_numbers_sim_churners_last1_top10	Count pairs similarity churners last month top 10
count_numbers_sim_churners_last1_topall	Count pairs similarity churners last month top all
count_numbers_sim_churners_last2_top3	Count pairs similarity churners last two months top 3
count_numbers_sim_churners_last2_top5	Count pairs similarity churners last two months top 5
count_numbers_sim_churners_last2_top10	Count pairs similarity churners last two months top 10
count_numbers_sim_churners_last2_topall	Count pairs similarity churners last two months top all
sum_duration_calls_sim_churners_last1_top3	Sum duration of calls similarity churners last month top 3
sum_duration_calls_sim_churners_last1_top5	Sum duration of calls similarity churners last month top 5
sum_duration_calls_sim_churners_last1_top10	Sum duration of calls similarity churners last month top 10
sum_duration_calls_sim_churners_last1_topall	Sum duration of calls similarity churners last month top all
sum_duration_calls_sim_churners_last2_top3	Sum duration of calls similarity churners last two months top 3
sum_duration_calls_sim_churners_last2_top5	Sum duration of calls similarity churners last two months top 5
sum_duration_calls_sim_churners_last2_top10	Sum duration of calls similarity churners last two months top 10
sum_duration_calls_sim_churners_last2_topall	Sum duration of calls similarity churners last two months top all
sum_num_calls_sim_churners_last1_top3	Sum number of calls similarity churners last month top 3
sum_num_calls_sim_churners_last1_top5	Sum number of calls similarity churners last month top 5
sum_num_calls_sim_churners_last1_top10	Sum number of calls similarity churners last month top 10
sum_num_calls_sim_churners_last1_topall	Sum number of calls similarity churners last month top all
sum_num_calls_sim_churners_last2_top3	Sum number of calls similarity churners last two months top 3
sum_num_calls_sim_churners_last2_top5	Sum number of calls similarity churners last two months top 5
sum_num_calls_sim_churners_last2_top10	Sum number of calls similarity churners last two months top 10
sum_num_calls_sim_churners_last2_topall	Sum number of calls similarity churners last two months top all
n_churners_last1_top3	Number of churners last month top 3
n_churners_last1_top5	Number of churners last month top 5
n_churners_last1_top10	Number of churners last month top 10
n_churners_last2_top3	Number of churners last two months top 3
n_churners_last2_top5	Number of churners last two months top 5
n_churners_last2_top10	Number of churners last two months top 10

Table B.2: Embedding features aggregated per month.

Bibliography

[1] Abdelrahim Kasem Ahmad, Assef Jafar, and Kadan Aljoumaa. Customer churn prediction in telecom using machine learning in big data platform. *Journal of Big Data*, 6(1):28, 2019.

[2] Antreas D Athanassopoulos. Customer satisfaction cues to support market segmentation and explain switching behavior. *Journal of business research*, 47(3):191–207, 2000.

[3] Michel Ballings and Dirk Van den Poel. Customer event history for churn prediction: How long is long enough? *Expert Systems with Applications*, 39(18):13517–13522, 2012.

[4] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pages 585–591, 2002.

[5] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

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

[7] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 891–900. ACM, 2015.

[8] Haochen Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. A tutorial on network embeddings. *arXiv preprint arXiv:1808.02590*, 2018.

[9] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[10] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, and Yuan Tang. Xgboost: extreme gradient boosting. *R package version 0.4-2*, pages 1–4, 2015.

[11] M Claesen and B De Moor. Hyperparameter search in machine learning. arxiv 2015. *arXiv preprint arXiv:1502.02127*.

[12] Koustuv Dasgupta, Rahul Singh, Balaji Viswanathan, Dipanjan Chakraborty, Sougata Mukherjea, Amit A Nanavati, and Anupam Joshi. Social ties and their relevance to churn in mobile telecom networks. In *Proceedings of the 11th international conference on Extending database technology: Advances in database technology*, pages 668–677. ACM, 2008.

[13] TL Oshini Goonetilleke and HA Caldera. Mining life insurance data for customer attrition analysis. *Journal of Industrial and Intelligent Information*, 1(1), 2013.

[14] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

[15] Montserrat Guillén, Jens Perch Nielsen, Thomas H Scheike, and Ana Maria Pérez-Marín. Time-varying effects in the analysis of customer loyalty: A case study in insurance. *Expert systems with Applications*, 39(3):3551–3558, 2012.

[16] Michael Haenlein. Social interactions in customer churn decisions: The impact of relationship directionality. *International Journal of Research in Marketing*, 30, 03 2013.

[17] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[18] Bingquan Huang, Brian Buckley, and T-M Kechadi. Multi-objective feature selection by using nsga-ii for customer churn prediction in telecommunications. *Expert Systems with Applications*, 37(5):3638–3646, 2010.

[19] Bingquan Huang, Mohand Tahar Kechadi, and Brian Buckley. Customer churn prediction in telecommunications. *Expert Systems with Applications*, 39(1):1414–1425, 2012.

[20] Yiqing Huang, Fangzhou Zhu, Mingxuan Yuan, Ke Deng, Yanhua Li, Bing Ni, Wenyuan Dai, Qiang Yang, and Jia Zeng. Telco churn prediction with big data. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 607–618. ACM, 2015.

[21] Shin-Yuan Hung, David C Yen, and Hsiu-Yu Wang. Applying data mining to telecom churn management. *Expert Systems with Applications*, 31(3):515–524, 2006.

[22] Joseph B Kruskal and Myron Wish. *Multidimensional scaling*, volume 11. Sage, 1978.

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

[24] Lizi Liao, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. Attributed social network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 30(12):2257–2270, 2018.

[25] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International Conference on Machine Learning*, pages 2113–2122, 2015.

[26] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[27] Teemu Mutanen, Jussi Ahola, and Sami Nousiainen. Customer churn prediction-a case study in retail banking. In *Proc. of ECML/PKDD Workshop on Practical Data Mining*, pages 13–19, 2006.

[28] Scott A Neslin, Sunil Gupta, Wagner Kamakura, Junxiang Lu, and Charlotte H Mason. Defection detection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of marketing research*, 43(2):204–211, 2006.

[29] Evelien Otte and Ronald Rousseau. Social network analysis: a powerful strategy, also for the information sciences. *Journal of information Science*, 28(6):441–453, 2002.

[30] Marcin Owczarczuk. Churn models for prepaid customers in the cellular telecommunication industry using large data marts. *Expert Systems with Applications*, 37(6):4710–4712, 2010.

[31] Benjamin Paaßen, Claudio Gallicchio, Alessio Micheli, and Alessandro Sperduti. Embeddings and representation learning for structured data. *CoRR*, abs/1905.06147, 2019.

[32] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[33] Carlos Andre Reis Pinheiro. *Social network analysis in telecommunications*, volume 37. John Wiley & Sons, 2011.

[34] Yossi Richter, Elad Yom-Tov, and Noam Slonim. Predicting customer churn in mobile networks through analysis of social groups. In *Proceedings of the 2010 SIAM international conference on data mining*, pages 732–741. SIAM, 2010.

[35] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[36] M Saravanan and GS Vijay Raajaa. A graph-based churn prediction model for mobile telecom networks. In *International Conference on Advanced Data Mining and Applications*, pages 367–382. Springer, 2012.

[37] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25:2951–2959, 2012.

[38] Reza Allahyari Soeini and Keyvan Vahidy Rodpysh. Applying data mining to insurance customer churn management. *Int. Proc. Comput. Sci. Inf. Technol*, 30:82–92, 2012.

[39] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[40] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[41] Chih-Fong Tsai and Yu-Hsin Lu. Customer churn prediction by hybrid neural networks. *Expert Systems with Applications*, 36(10):12547–12553, 2009.

[42] Dirk Van den Poel and Bart Lariviere. Customer attrition analysis for financial services using proportional hazard models. *European journal of operational research*, 157(1):196–217, 2004.

[43] Wouter Verbeke, Karel Dejaeger, David Martens, Joon Hur, and Bart Baesens. New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218(1):211–229, 2012.

[44] Wouter Verbeke, David Martens, and Bart Baesens. Social network analysis for customer churn prediction. *Applied Soft Computing*, 14:431–446, 2014.

[45] Wouter Verbeke, David Martens, Christophe Mues, and Bart Baesens. Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert systems with applications*, 38(3):2354–2364, 2011.

[46] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[47] Guo-en Xia and Wei-dong Jin. Model of customer churn prediction on support vector machine. *Systems Engineering-Theory & Practice*, 28(1):71–77, 2008.

[48] Yaya Xie, Xiu Li, EWT Ngai, and Weiyun Ying. Customer churn prediction using improved balanced random forests. *Expert Systems with Applications*, 36(3):5445–5449, 2009.