AN ENSEMBLE CLASSIFICATION MODEL FOR DETECTING VOICE PHISHING IN TELECOMMUNICATION NETWORKS AND ITS INTEGRATION INTO A VISUAL ANALYSIS TOOL

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

By

Hüseyin Eren Çalık

September 2022

AN ENSEMBLE CLASSIFICATION MODEL FOR DETECTING VOICE PHISHING IN TELECOMMUNICATION NETWORKS AND ITS INTEGRATION INTO A VISUAL ANALYSIS TOOL

By Hüseyin Eren Çalık

September 2022

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Uğur Doğrusöz(Advisor)

Faruk Polat

Abdullah Ercüment Çiçek

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

AN ENSEMBLE CLASSIFICATION MODEL FOR DETECTING VOICE PHISHING IN TELECOMMUNICATION NETWORKS AND ITS INTEGRATION INTO A VISUAL ANALYSIS TOOL

Hüseyin Eren Çalık
M.S. in Computer Engineering
Advisor: Uğur Doğrusöz
September 2022

Voice phishing, a method of social engineering fraud performed over phone calls, has been a major problem globally since the use of phones became widespread. Traditional and modern methods to detect these fraud schemes include visual analysis of the customers' behaviour, rule-based systems and machine learning models such as clustering, decision trees, shallow classifiers and deep learning models. Visual analysis depends only on human expertise and requires very high labor force to be effective. Rule-based systems are useful for extreme cases but are vulnerable to concept drifts. The-state-of-the-art methods generally utilize machine learning approaches. However, they require one or more of feature engineering done by experts, high computational power and privacy infringements. Therefore, in collaboration with Turkcell Technology, we aimed to develop a system that benefits from the advantages of the traditional methods while exploiting the effectiveness and efficiency of the state-of-the-art ones to tackle this issue. In doing so, we integrated an ensemble learning model to an existing visualization tool for detecting fraud users. This tool visualizes relational data as knowledge graphs, shows the informational data as texts and statistical data with charts and texts. Our ensemble learning model has two deep neural networks and one decision tree classifier. Multiple neural networks are used to reduce the variance and make a more stable model. One of them is composed of an input layer, two hidden layers with 200 nodes using Rectified Linear Unit (ReLU) activation function, each followed by a dropout layer and an output layer of one node with sigmoid activation function. We used dropout layers in this network to prevent over-fitting. The second neural network we built has 3 hidden layers instead with node numbers 64, 64 and 32, respectively, with ReLU as their activation function. To feed these models, a total of 34 features, 20 of which are raw, have been

engineered with Turkcell fraud experts. The aggregation of the outputs is done by taking their average. We measured the success of our model by calculating the F1 Score as the class imbalance is high. Our model's F1 score is 0.82 with a precision of 0.82 and a recall of 0.83. Also, with the integration of our model into this visualization tool, a framework was formed allowing mobile network operators to examine and detect fraud cases more efficiently and act accordingly.

Keywords: Telecommunication networks, call graphs, visual network analysis, graph visualization, fraud detection, machine learning, deep neural networks, deep learning.

ÖZET

TELEKOMÜNKASYON AĞLARINDA SESLİ OLTA SALDIRILARININ BİRLEŞİK SINIFLANDIRMA MODELİ İLE TESPİTİ VE GÖRSEL ANALİZ ARACINA ENTEGRASYONU

Hüseyin Eren Çalık
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Uğur Doğrusöz
Eylül 2022

Telefon görüşmeleri üzerinden gerçekleştirilen bir sosyal mühendislik dolandırıcılığı yöntemi olan sesli kimlik avı, telefon kullanımının yaygınlaşmasından bu yana dünya çapında büyük bir sorun haline geldi. Bu sahtekarlıkları tespit etmek için geleneksel ve modern yöntemler şu şekilde listelenebilir; müşteri davranışlarının görsel analizi, kural tabanlı karar sistemleri, kümeleme algoritmaları, karar ağaçları, sığ sınıflandırıcılar ve derin öğrenme modelleri gibi makine öğrenme modelleri. Görsel analiz sadece insan uzmanlığına bağlıdır ve etkili olabilmesi için çok yüksek iş gücü gerektirir. Kural tabanlı sistemler uç durumlar için kullanışlıdır ancak konsept kaymalarına karşı savunmasızdır. En gelişmiş yöntemler genellikle makine öğrenimi yaklaşımlarını kullanır. Ancak, makine öğrenmesi yöntemlerinin, uzmanlar tarafından yapılan özellik mühendisliği çalışması ve yüksek hesaplama gücü gibi gereklilikleri ve gizlilik ihlalleri gibi kısıtları vardır. Bu nedenle Turkcell Teknoloji ile ortaklaşa çalışarak, bu sorunu çözmek için geleneksel yöntemlerin avantajlarından ve son teknoloji yöntemlerin etkinlik ve verimliliğinden yararlanan bir sistem geliştirmeyi hedefledik. Dolandırıcı kullanıcıları tespit etmek için bir birleşik öğrenme modelini mevcut bir görselleştirme aracına entegre ettik. Bu araç, ilişkisel verileri bilgi çizgeleri olarak görselleştirirken, bilgileri ve istatistiksel verileri çizelgeler ve metinler olarak gösteren bir araçtır. Birleşik öğrenme modelimiz iki derin sinir ağı ve bir karar ağacı sınıflandırıcısından oluşturmaktadır. Varyansı azaltmak ve daha kararlı bir model oluşturmak için iki farklı yapay sinir ağından faydalandık. İlk yapay sinir ağı; bir giriş katmanından, Rektifiye Edilmiş Doğrusal Birim (ReLU) aktivasyon fonksiyonunu kullanan 200 yapay nöron içeren iki gizli katmandan, gizli katmanları takip eden nöron seyreltme katmanlarından

ve sigmoid aktivasyon fonksiyonlu bir yapay nöron içeren çıkış katmanından oluşmaktadır. Aşırı uyumlamayı önlemek için bu ağda nöron seyreltme katmanları kullanılmıştır. Oluşturduğumuz ikinci sinir ağı, aktivasyon fonksiyonu ReLU olan, sırasıyla 64, 64 ve 32 yapay sinir hücresi sahip 3 gizli katmana sahiptir. Bu modellerde kullanmak için Turkcell dolandırıcılık uzmanlarıyla birlikte 20'si ham olmak üzere toplam 34 özellik tasarlanmıştır. Son sonuçların hesabı, modellerin çıktılarının ortalamaları alınarak yapılmıştır.

Modelimizin başarısı, sınıf dengesizliği yüksek olduğu için F1 skoru ile ölçülmüştür. Modelimiz 0.82 kesinlik değeri, 0.83 duyarlılık değeri ve 0.82 F1 skoruna sahiptir. Ayrıca modelimizin görselleştirme aracına entegrasyonu ile mobil şebeke operatörlerinin dolandırıcılık olaylarını daha etkin bir şekilde inceleyerek buna göre hareket etmelerini sağlayan bir çerçeve oluşturulmuştur.

Anahtar sözcükler: Telekomünikasyon ağları, çağrı çizgeleri, görsel ağ analizi, çizge görselleştirme, dolandırıcılık tespiti, makine öğrenmesi, derin sinir ağları, derin öğrenme.

Acknowledgement

First of all, I would like to express my deepest gratitude and my utmost respect to my advisor, Prof. Dr. Uğur Doğrusöz for his support, motivation and patience. I believe his principles are what a successful student or academician need. I hope to be more like him in the future.

I would like to thank Prof. Dr. Fazlı Can and Prof. Dr. Özcan Öztürk for their support throughout my masters. I would like to thank Doç. Dr. Ercüment Çiçek and Doç. Dr. Shervin Arashloo for their help in our work. I would also like to thank Prof. Dr. Faruk Polat for being in my defence jury and providing valuable feedback.

I would like to thank Turkcell team for making this work possible.

I would also like to acknowledge The Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing me financial support for this thesis within the scope of TEYDEB 1505 project (in Bilkent-Turkcell collaboration) with grant number 5200049 and TEYDEB 1505 project (in Bilkent-Huawei collaboration) with grant number 5180088.

Thanks to my valuable friends Sina Barazandeh, Yusuf Sait Canbaz, Mustafa Can Çavdar, Musa, Pouria, Pouya, Sepehr, Asma and many more for making this difficult journey fun for me. I would also like to thank all the faculty members, graduate students and staff in the CS Department who made my day, even if it was just by saying hello.

Special thanks to Hasan Balcı who was like a brother to me throughout whole process, for his guidance, for reminding me many things I tend to forget easily and of course for the extreme fun times we had.

Special thanks to Funda Tan for supporting me and motivating me to finish my thesis.

Last but not least, I would like to express my deepest gratitude to my mother and father, Sare and Temel Çalık, for their unconditional love, help and support. I love and appreciate you. I also felt all the support from my extended family, I thank each and every one of them that they were always beside me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Phone calls have become a permanent part of our lives. Although there has been a sharp increase in video calls and other digital communication methods, number of phone calls is not declining; in fact, it is increasing and expected to increase further. According to a recent report [5] of a global spam filtering company where 1800 businesses and 12000 customers and 151 billion calls were involved, phone calls are the first choice of the businesses to communicate with their customers and it is also the consumers' first preference. However, the same report shows that spam and fraud calls are growing in number and has global effects. As fraud may greatly harm consumers, businesses and telecom companies put more effort in detecting and preventing these calls. The most used method of detection and prevention is blacklisting and many telecom companies either use this technique themselves or enter into a partnership with companies that provide this service. While blacklisting is a good method for preventing some types of spam/fraud calls, the fraudsters constantly change their numbers and their methods. Also, as it is a user-dependent method, it can be difficult to detect fraudsters before they cause some harm.

Same report shows that among these types of fraud calls, type that contain impersonation is the most prevalent one, as 62% of the survey respondents had

received at least one call where the caller impersonates a business or a government officer. This type of fraud, called *voice phishing*, is difficult to prevent with blacklisting, as fraudsters are able to change their numbers easily and use fake information to have their sim cards. Therefore, to prevent this type of fraud rapidly, telecom companies started to utilize behaviour analysis of their customers. The purpose of behaviour analysis is to detect anomalies and investigate the anomalous users.

These methods include Visual Analytics, Cluster Analysis, Rule-based Classification and Statistical Classification methods such as Linear Classifiers, Decision Trees and Neural Networks. Although some of these methods are newer and more successful, all of them have weaknesses in detecting Voice Phishing.Visual analytics need high labour force yet allows a thorough analysis. Cluster analysis can provide an automated system, whereas, the results can bear meaning only to an expert on both machine learning and fraud schemes. Statistical classification methods can be efficient and effective; however, they need precise feature engineering and can be vulnerable to changes in fraudsters behaviours.

1.1 Contribution

In this thesis, we present the integration of an ensemble learning model that combines two classification methods, deep neural networks (DNN)[6] and decision trees[7], into a visual analysis tool, Hydra Telecom[8]. We also provide features that were engineered by reviewing the literature of voice phishing and voice phisher behaviour and working with Turkcell domain experts, which can be used in other applications. To the best of our knowledge, this work is the first approach that integrates a machine learning model into a visual analysis tool, letting domain experts use model's predictions to detect fraudsters efficiently, assessing the results of the ensemble classifier model and improving the results by marking the suspected phone numbers via a visual analysis tool.

We have organised the rest of this paper in the following way: Chapter 2 gives

background information of the problem and the methods with their examples. In Chapter 3, we explain our model and in Chapter 4, evaluation of our model is presented. Chapter 5 concludes the thesis with a discussion of our results and possible feature work.

Chapter 2

Background and Related Work

2.1 Voice Phishing

Voice phishing or vishing [9][10] is a type of social engineering fraud where the fraudsters use phone calls to deceive victims for financial gain. Their method can be basically formulated as follows: introducing themselves as someone of authority and power, telling fake scenarios in which victims are in troublesome situation, imposing urgency and requesting sensitive information or an act that is harmful for the victim but beneficial for the fraudster. They pretend to be or impersonate people from professions such as law enforcement, government officials, bank employees or tech supports. Their targets can be randomly selected individuals, vulnerable people or people they already have personal information about, gained with other social engineering techniques or data breaches.

An example scheme can be illustrated as:

- A fraudster buys information from a company or another criminal that contain names, phone numbers and ages of a particular group of people.
- Fraudster impersonates a bank manager and calls people on his/her list one by one.

- Fraudster tells the victims that they have high amounts of debt because of some error.
- Fraudster tells that if the victims do not take action in a short period of time, they will be supposed to pay those high amounts.
- Fraudster asks for money or account information of the victims, telling that s/he will solve the problem.

2.2 Machine Learning

Machine learning is a branch of Artifical Intelligence that studies development and analysis of the systems that use data to improve performance on a set of tasks [11]. Machine learning algorithms construct a model based on training data to generate predictions or conclusions without explicit instructions contrary to traditional Artificial Intelligence methods. Therefore, in applications such as medicine, email filtering, speech recognition and computer vision, where it is difficult or impractical to create traditional Artificial Intelligence algorithms to do the required tasks, machine learning techniques are utilized.

A machine learning model's primary purpose is to generalize from the training data it is provided with. In other words, the model should generate predictions or conclusions accurately on unlearned or unobserved examples after being exposed to the training examples.

Machine learning is generally divided into three categories:

- Supervised learning: Supervised learning algorithms construct a mathematical model of a data set that includes both the inputs and expected outcomes. Therefore to build a supervised learning model, each training example must consist of one or more inputs and the expected output. Each training example in the mathematical model is represented by an array or vector, sometimes known as a feature vector, and the training data is

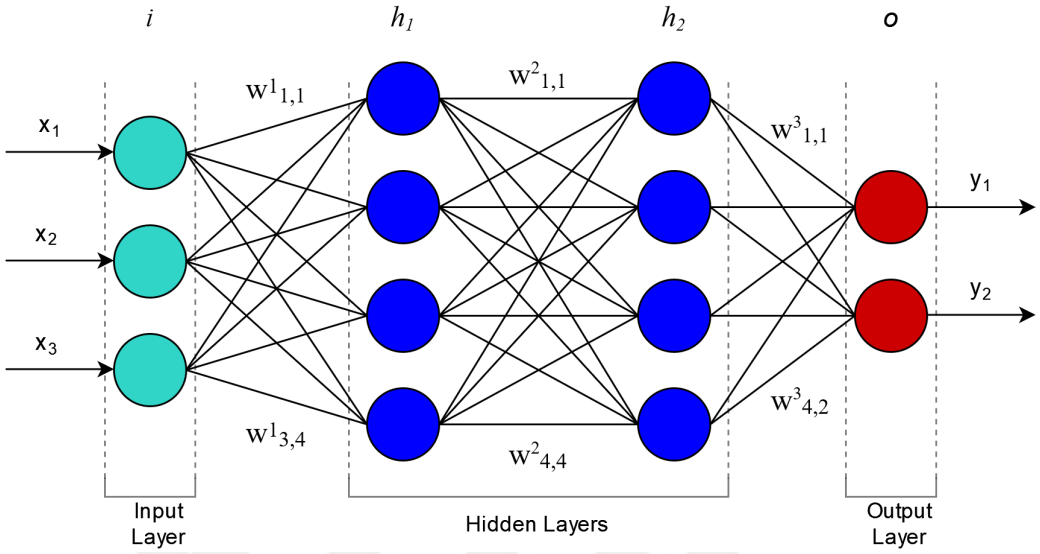
represented by a matrix. By optimizing an objective function iteratively, supervised learning algorithms discover a function that may be used to predict the output associated with a new set of inputs that were not included in the training data.

- Unsupervised learning: Unsupervised learning techniques use a collection of unlabeled, unclassified, and uncategorized training data containing only the inputs to discover a structure in the data, such as groups or clusters. Clustering and dimensionality reduction can be given as examples to unsupervised approaches.

- Reinforcement learning: A reinforcement learning program directly learns to give sequences of predictions in an environment in which it must achieve uncertain, complex goals (such as playing a car race game against opponents). As the learning process continues, the program is given feedback, rewards or penalties, which the program tries to maximize or minimize.

2.3 Artificial Neural Networks and Deep Learning

Artificial neural networks are computer systems inspired by the neural networks that make up animal brains. An artificial neural networks is comprised of a network of interconnected units or nodes known as artificial neurons, which resemble the neurons of a biological brain. Similar to the synapses in a human brain, each connection, or link, provides the communication between two neurons. However, artificial neural networks are bipartite graphs and have a layered organization. Therefore, there can be no link between two neurons of the same layer.



Each artificial neuron contains inputs and outputs that can be used by the next layer of artificial neurons. The inputs can either be the features of the data sample, or the outputs of other neurons from the previous layer. The outputs of the neurons of the final layer are the output of the system. The initial inputs are external data features, such as vectorized images or handcrafted features. Weighted sum of all inputs, weighted by connections to the neuron, is passed through a nonlinear activation function to calculate the output of a neuron. The final outputs are the result that complete the task, such as object recognition or fraud detection.

The network's weights are changed to make the result more accurate in the learning process. The learning process is comprised of defining a loss function, or a cost function, that is evaluated at regular intervals and changing the weights to minimize the output of this loss function. The loss function is basically the estimation of the model's error in prediction. Therefore, adjusting the weights to minimize the loss function makes the predicted result closer to the actual result.

For minimizing the loss functions, thereby training neural networks, gradient descent is the most practiced algorithm. Gradient descent can be denoted as follows. If a multi-variable function $F(x)$ is differentiated at point i, then $F(x)$ rapidly decreases from point i to $i - \eta \nabla F(i)$, where η is a sufficiently small step

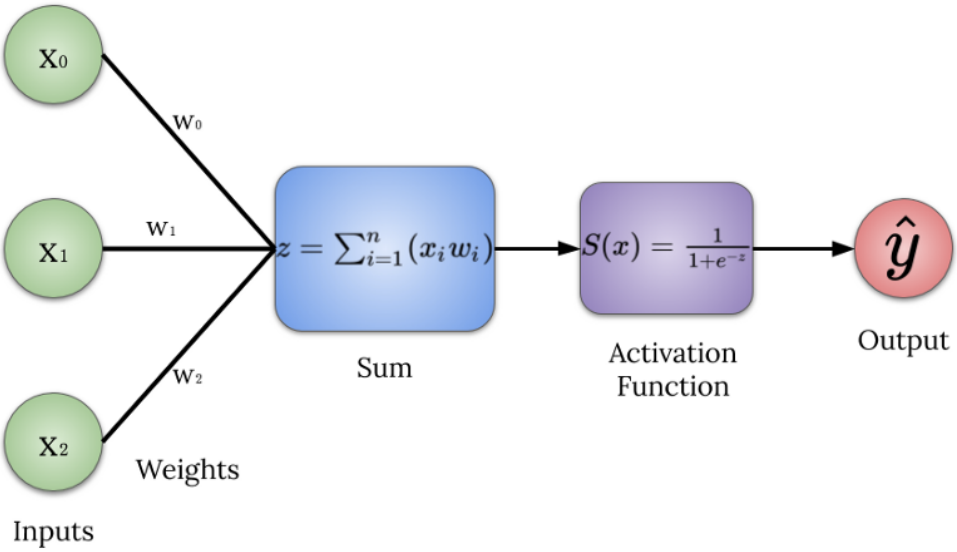


Figure 2.1: Illustration of an artificial neuron[1].

size or learning rate. Therefore, for the loss function $F(x)$ and with a sufficiently small learning rate, $F(i) > F(i - \eta \nabla F(i))$.

However, in a deep neural network, there can be layers with nonlinear activation functions. Thus, calculating the gradient in such networks is more complex. For this calculation, the backpropagation algorithm is used. Backpropagation computes the gradient of the loss function layer by layer. It starts from last layer and iterates backwards by computing the gradients with the use of chain rule.

2.4 Data Visualization

Data visualization is the depiction of using visuals such as different types of charts, plots and histograms [12]. These informational visualizations simplify the complicated data linkages and data-driven conclusions. Therefore, it can be stated that the purpose of data visualization is to develop methods for presenting abstract information in a comprehensible manner. It is being used as a fundamental component of scientific research, data analysis and data mining.

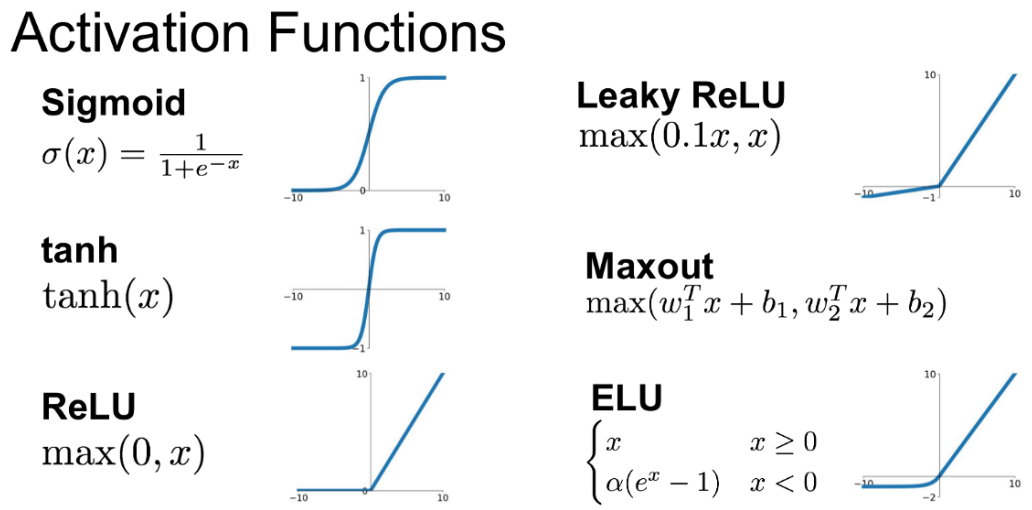


Figure 2.2: Plots and equations of some commonly used activation functions[2].

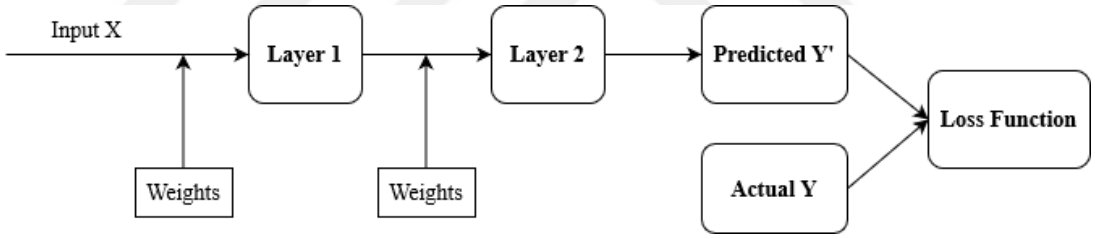


Figure 2.3: Illustration of a loss function's modus operandi.

2.4.1 Graph Drawing

Graph drawing or layout of a graph is creating a visual representation of a graph with mathematical and data visualization techniques. A drawing or a layout of a graph is the graphical depiction of its vertices, or nodes, and edges, or links. Vertices can be simple graphical shapes, images or pictures; edges can be straight or curved lines or simple polygonal chains depending on the application and the purpose of the visualization. Direction of the directed edges can be denoted with arrowheads.

The layout process is principally placing the vertices and edges between. However, there can be many different layouts of one graph. This is caused by the fact that the problems related to the deterministic graph layouts are NP-complete

problems and to make real life applications feasible, mostly randomized approximation algorithms are used as heuristics. The algorithms using these techniques calculate the positions of nodes and connection points of the edges to provide sufficiently good layouts. As there is no method for creating a perfect layout, there are criteria that are generally accepted that are used to evaluate the quality of a layout [13]. These criteria aim to provide results which are highly understandable while being aesthetically pleasing. They are as follows:

- Minimum number of node overlaps: Overlapping nodes can easily complicate the graph and severely impair the understandability of the graph. Therefore, it is considered as a crucial criterion.
- Minimum total area of the layout: Total area should be minimum as this will define the zoom level, visibility and understandability of the entities and relations.
- Layout symmetry: Symmetric drawing provides efficient use of the layout area and better understandability of the graph.
- Minimum number of edge crossings: Edge crossings may cause confusion; therefore, they should be optimally low.
- Optimal edge lengths: This criterion can be divided to two sections. Total length and maximum of the edge lengths should be minimum while preserving the understandability of the graph. Secondly, the edge lengths should be uniform.

There are well-known techniques that aim to give good results based on these criteria while retaining low time complexity. Examples of these techniques are:

- Force-directed graph layouts [14] provide drawings with minimum possible number of overlapping nodes and edge crossings with uniform edge lengths. Generally, the nodes are simulated as electrically charged particles that repulse each other and edges as springs with attractive forces. The layout iteratively continues until the system reaches an equilibrium state.

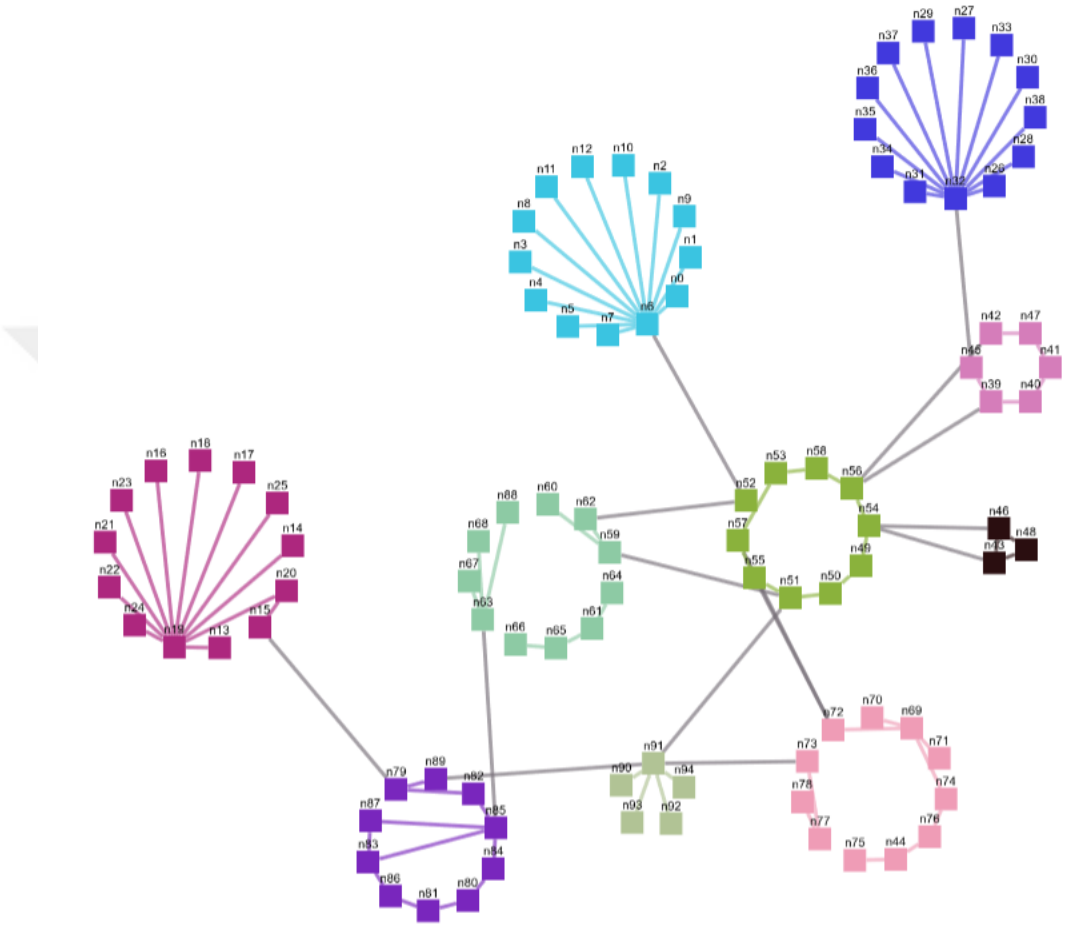


Figure 2.4: A graph visualized with CiSE [3], a circular spring embedder layout algorithm.

- Circular layouts [15][16] determine the position of the nodes on circles while minimizing the edge crossings. The minimization of the edge crossings of a circular layout is an NP-complete problem, hence, heuristic methods are used.

- Orthogonal layout [16] methods position edges horizontally or vertically parallel to the the coordinate axes.

- Spectral layout [17] methods position the nodes, treating the layout space as the Cartesian coordinate system, based on the eigenvectors of the Laplacian

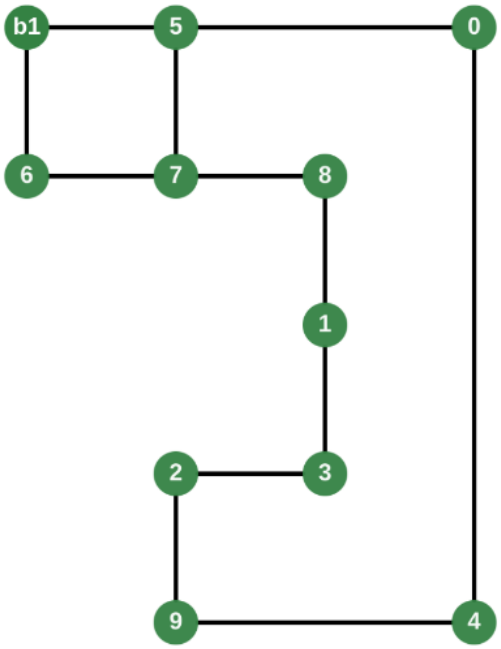


Figure 2.5: A graph visualized with an orthogonal layout algorithm [4].

derived from the adjacency matrix of the graph.

- Hierarchical layout [18] methods provide layered layouts, vertically or horizontally, where the nodes of the same layer are positioned with same x or y values with respect to the chosen scheme.

2.5 Related Work

2.5.1 Visual Analytics

Kenneth Cox et al. [19] suggest that telecommunication fraud can be found through both group and individual analysis. They do this by using graphs, bar charts, and line charts. This is one of the first pieces of work that uses visual analytics techniques to find fraud. The main idea is to make visual interfaces that let the data be explored. Clustering techniques are used to improve the way that nodes and edges can be seen when they interact.

Becker et al .[20] use line plots to compare different user profiles and find strange behaviors. The authors also explain how different types of fraud in the field can be found with the aid of visual analytics models.

2.5.2 Cluster Analysis

In the work of Hilas et al. [21], it is intended to show how some well-known techniques of unsupervised learning can be used on telecommunications data to provide information about how both legitimate users and fraudsters act. In their work, raw usage data is processed into user profiles. Based on their analysis, it can be stated that, in terms of user profile creation, a user's behaviour collected through one week can be especially discriminative, whereas data collected through one month is less useful.

Their results demonstrate that misclassification can happen as a result of mixed behavioral types. In other words, there are instances in which a valid user behaves like a fraudster, for instance by making lengthy or costly calls, while a fraudster may behave like a regular user, for instance by being more restrained and attempting to imitate genuine user behavior. Thus, the clustering technique would be inefficient if it is forced to build two unique data groups, one with genuine usage and one with fraudulent usage.

It can be concluded that their observations show that clustering alone cannot be used to decide whether or not a user account is being used to commit fraud. Hence, information from other methods is also needed.

2.5.3 Decision Tree Approach

In the work titled "An Application of Decision Trees for Rule Extraction Towards Telecommunications Fraud Detection" [22], decision trees are used to discover thresholds between normal and fraudulent user behavior in a telecommunications network. The authors apply feature selection methods for building the decision

tree. It is stated that the specified thresholds depend on the training data and the thresholds may be adjusted as more fraud cases are used. The thresholds are stated to be used as a clue but not as a proof of fraud. The presented methodology claimed to be applied to similar problems in a mobile or data network. However, since rules often depend on the specific nature of each domain, the results can be misleading if the findings are directly generalized to similar applications as it is stated by the authors.

2.5.4 Deep Learning Approach

Chouiekh et al. [23] compares the effectiveness of convolution neural networks with classic machine learning techniques to predict fraudulent events in telecom networks. Their accuracy of 82 percent indicates that DCNN architecture produced the best results, surpassing Support Vector Machine, Random Forest, and Gradient Boosting Classifier algorithms. Also, this approach achieves the optimum performance in terms of training time compared to the methods they evaluated their model with, allowing the telecom operators to cut costs associated with fraudulent usage of services. However, they do not compare their method with other deep architectures and the descriptions of the schemes they target are ambiguous.

Chapter 3

An Ensemble Machine Learning Approach

Our approach consists of three main steps. In the first step, we determined features which are aimed to signify the presence of voice phishing fraud. Although they are engineered particularly for voice phishing type of fraud, they are also expected to be useful in similar telecommunication fraud detection applications. In the second step, we created a combined model consisting of two feed forward artificial neural networks and a decision tree classifier. We trained and tested our models using the features we defined in the first step. In the third and final step, to use the results from our ensemble learning model in the visual analysis software tool Hydra Telecom, we added a custom query to the tool. This query's function is reading the results of the machine learning model, retrieving the relevant information on suspicious cases and presenting them both visually and as a listing.

The motives behind choosing such an ensemble model are improving the results and making the model optimally adaptive to the changes in data. First DNN's individual testing precision and recall scores are 0.8154 and 0.8153, second DNN's testing precision and recall scores are 0.8257 and 0.8348 whereas decision tree's testing precision and recall scores are 0.8296 and 0.8792. Although decision tree

appears to perform better, it is vulnerable to noisy data and relatively very reactive to the changes in data. As when this system is deployed, data with much bigger sizes will be used in each execution, it is strongly expected that our decision tree will not have rules inclusive for all cases of fraud present in such data. This makes it difficult to trust sharp rules provided by the decision tree. Therefore, we coupled it with two DNNs to have a more stable model while partially preserving the advantage of decision trees on the concept drift problem. The visualization of the decision tree also can provide insight on the data and its changes.

In the remainder of the chapter, we first discuss the data set used in our approach followed by the aforementioned steps of the approach.

3.1 Dataset Description

The dataset esentially consists of information of approximately 100,000 users and 110,000 cell phone numbers, or Mobile Station International Subscriber Directory Numbers (MSISDNs), and 75,000,000 call detail records (CDRs). CDRs have the information of a 70-day period. It was provided by Turkcell Technology and for privacy protection, all personally identifiable information, sensitive or non-sensitive, and other non-statistical data were masked. We removed corporate customers from the data as their probability of committing Voice Phishing fraud is considered to be extremely low. We used 72%, 8% and 20% of the data for training, validation and testing, respectively.

Twenty columns of the raw data are used as features, which include:

- 6 separate values for the amounts of last 3 months' bills and information on if they are paid or not,
- Billing city,
- Citizenship status,

- Date of subscription and other information that are necessary for the company's internal structure.

Each non-numerical information was mapped to an integer value to be used in the machine learning model.

3.2 Feature Engineering

As a result of our interviews with the domain experts, literature review and visual analysis of the data, we defined a total of 14 features that could be extracted using statistical and arithmetic operations. These features are as follows:

- Feature: Total number of MSISDNs

  Description: Total number of MSISDNs the owner currenly has or had in the past.

- Feature: Average activation time distance

  Description: Average time between activation dates of the MSISDN.

  Importance: If there are more than one MSISDN owned by the owner of the line, they tend to be activated in a relatively short period of time in fraud cases.

- Feature: Total number of IMEIs

  Description: Number of different International Mobile Equipment Identity (IMEI) numbers this MSISDN is used with.

- Feature: Number of fraud MSISDNs

  Description: Number of MSISDNs owner had whose other MSISDNs has been involved in fraud before. Plot of average values are shown in Figure 3.1.

- Feature: Number of fraud IMEIs

  Description: Number of IMEIs used with this MSISDN that has been involved in fraud before. Plot of average values are shown in Figure 3.2.

- Feature: MSISDNs called

  Description: Number of unique MSISDNs called.

  Importance: This feature is used in many anomalious behaviour detection applications.

- Feature: MSISDNs called

  Description: Number of unique MSISDNs that has called an MSISDN of interest. Plot of average values are shown in Figure 3.3.

- Feature: Parts of day call frequency

  Description: Frequency of calls in different parts of the day. We divided the day into four parts: 06.00AM to 09.00AM, 09.00AM to 07.00PM, 07.00PM to 01.00AM, 01.00AM to 06.00AM and calculated the ratios of calls made at each part to the number of total calls.

- Feature: Parts of week call frequency

  Description: Frequency of calls in different parts of the week. Similar to the parts of the day, we divided the week into two parts, work days and weekend.

- Feature: Call direction ratio

  Description: Ratio of outgoing/incoming calls.

  Importance: A very high or very low value may signal anomalious behaviour.

- Feature: Most used cell ratio

  Description: The ratio of the number of calls made from the cell, over which the highest number of calls made, to the number of all calls made.

  For example, if an MSISDN makes 20, 30 and 50 calls from cells with ID 1,2 and 3, respectively;

Number of calls made from the cell with ID 3 (from the cell through which it makes the most calls: 50

Total number of calls: 100;

The value we will use is: $50/100 = 1/2$.

- Feature: Average call duration.
- Feature: Standard deviation of call durations.
- Feature: Average number of calls made per day.

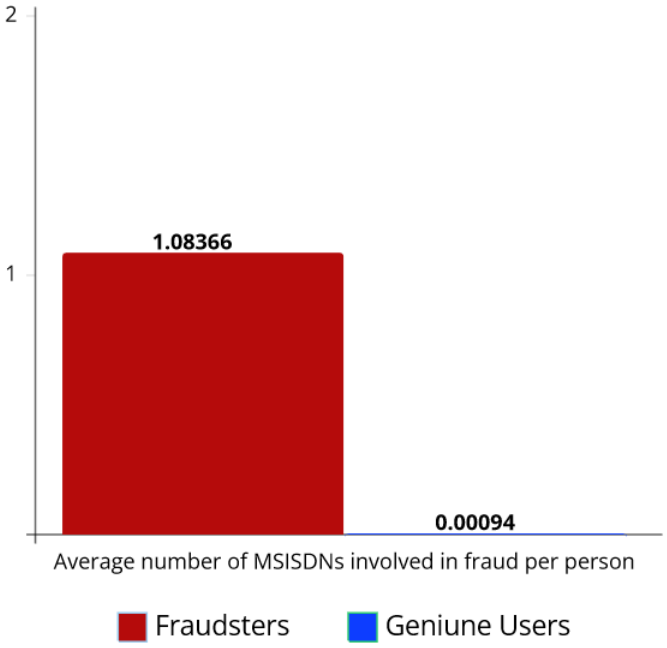


Figure 3.1: Users whose MSISDNs are involved in fraud tend to involve in fraud with other MSISDNs.

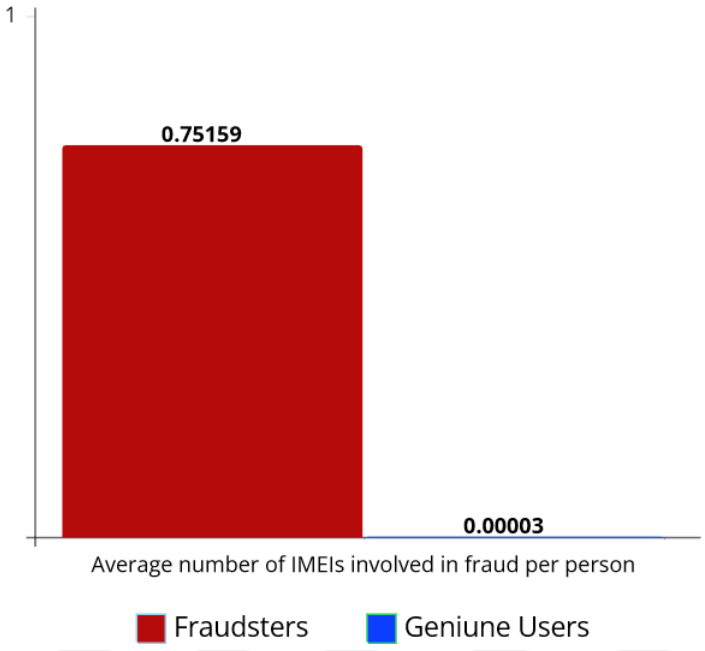


Figure 3.2: Fraudsters tend to use different IMEIs with MSISDNs that have been involved in fraud.

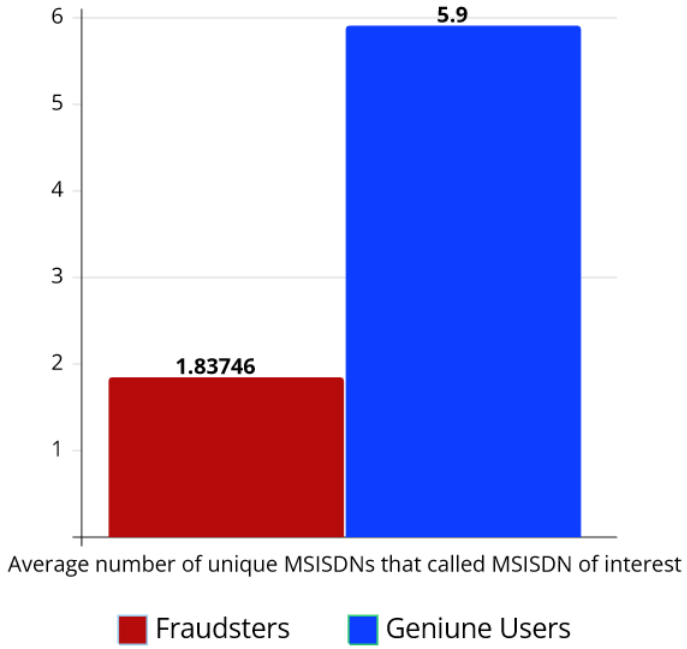


Figure 3.3: Average number of unique MSISDNs that called the MSISDN of interest is significantly low in fraud MSISDNs

We also defined other features that might be useful; however, we did not have the necessary data to make use of these. These include:

- Numerical similarity between the MSISDNs called: In some cases the fraudster randomly calls numerically consecutive MSISDNs. However, because our data was hashed, we could not make such analysis.
- Geographical analysis between the locations of MSISDNs called: Proximity of locations may provide information on genuinity of the calls.

3.3 Ensemble Learning Model

Our ensemble model is comprised of two DNNs and a decision tree classifier. First DNN consists of the input layer with 36 neurons for 36 numerical values, a total of 3 hidden layers, first two of which with 64 neurons and last one with 32 neurons and the output layer which has one neuron as shown in the Figure 3.5. The neurons in the hidden layers have Rectified Linear Unit (ReLU) activation function whose formula is: $f(x) = max(0, x)$ where x is the input of the neuron. Plot of ReLU function can be seen in Figure 3.4.

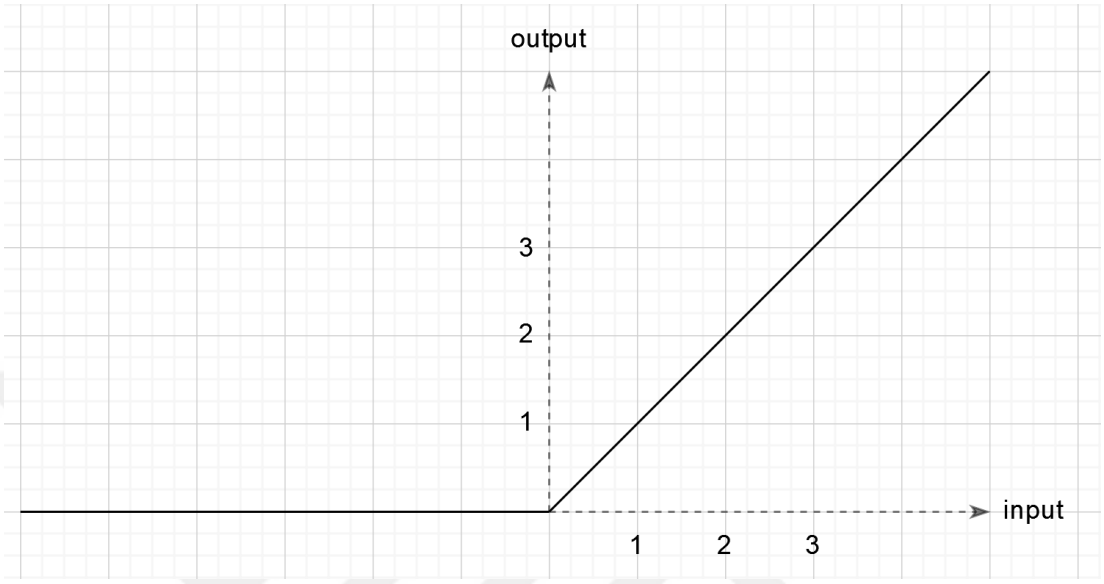


Figure 3.4: Plot of the ReLU function.

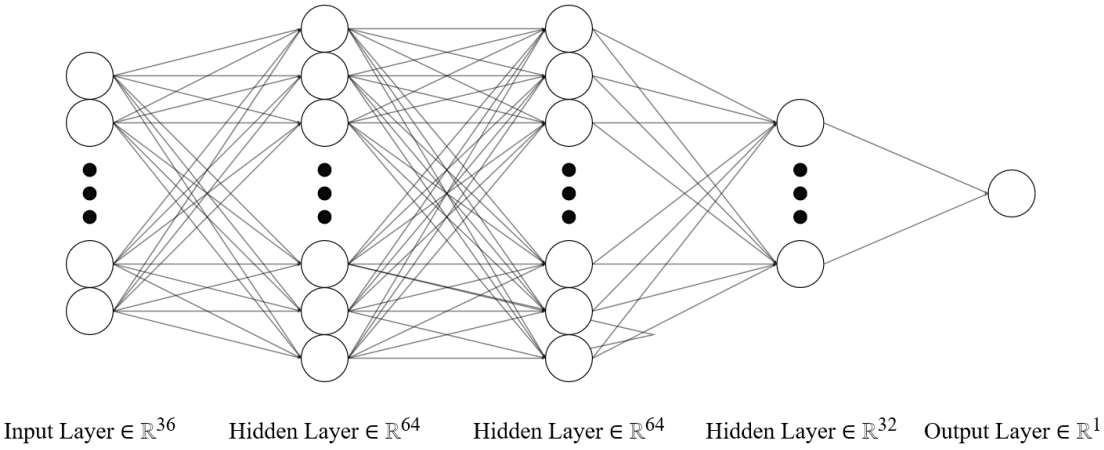


Figure 3.5: First DNN model.

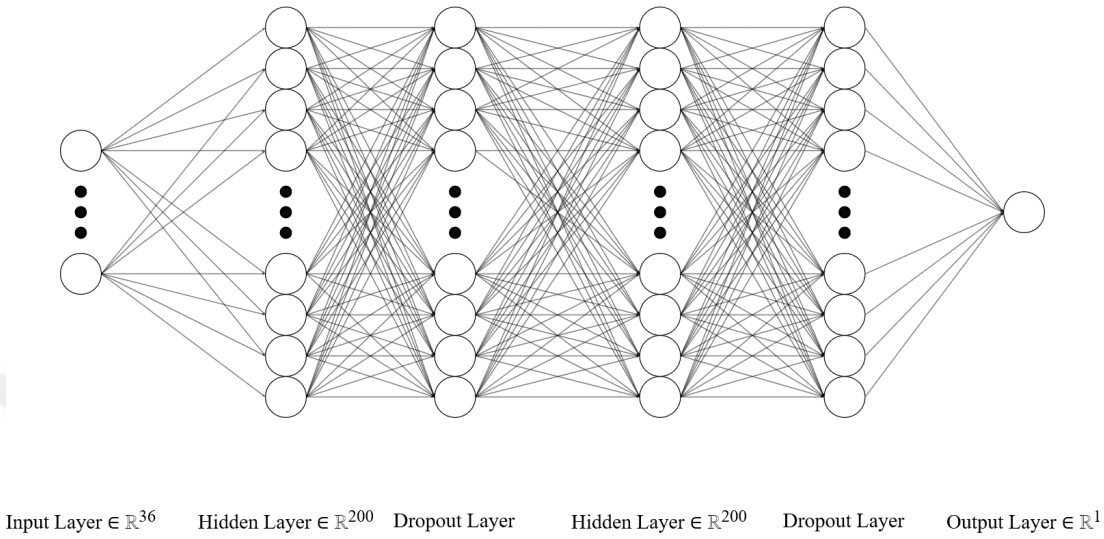


Figure 3.6: Second DNN model.

Second DNN's input layer is also composed of 36 neurons. However, it has a total of 2 hidden layers, each containing 200 neurons with ReLU. The output layer has one neuron with the sigmoid function. Sigmoid function maps the values between 0 and 1 in a non-linear fashion, as shown in the Figure 3.7. This perceptron also has a dropout applied for each hidden layer. The visual representation of the second model is shown in Figure 3.6.

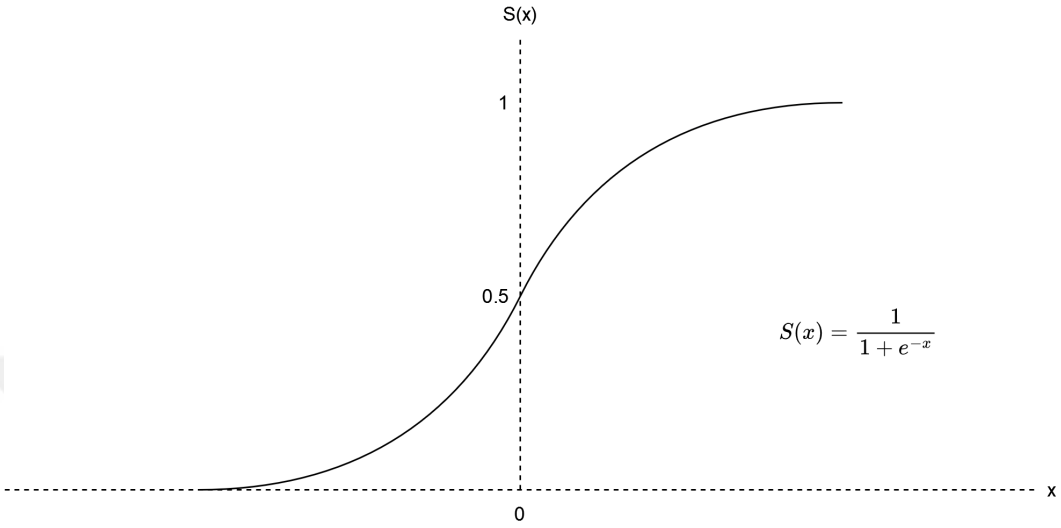


Figure 3.7: Plot of the sigmoid curve.

We rescaled our data using min-max normalization. This is a method used to scale features into a target range, between 0 and 1 in our case, to prevent domination of individual features and have a higher convergence rate. We also used dropout technique in the second DNN because it has a high number of neurons relative to the size of the input layer and we used a higher epoch number to train it, which may cause over-fitting without the regularization techniques such as dropout. Also, for the optimization, we used Adaptive Gradient Algorithm, which is an extended version of stochastic gradient descent algorithm, in both models. Their hyperparameters are determined experimentally using the training and validation set.

The hyperparameters of the first DNN are:

- $\alpha : 10^{-5}$

  Alpha is the learning rate parameter. 8 values from 10^{-1} to 10^{-8} were tried.

- $\beta 1 : 0.9$

  20 values from 0.6 to 0.9 were tried.

- $\beta 2 : 0.999$

  20 values from 0.699 to 0.999 were tried.

- $\epsilon : 10^{-7}$

  Epsilon is a small number that helps solve the division by zero problem. Therefore, the default values were directly used.

- *Batch size* : 38

  200 values, from 1 to 200 were tried.

- *Epoch number* : 175 In total of 13 values, from 1 to 300, that are divisible by 25 except 1.

The hyperparameters of the second DNN are:

- $\alpha : 10^{-3}$

  8 values from 10^{-1} to 10^{-8} were tried.

- $\beta 1 : 0.9$

  20 values from 0.6 to 0.9 were tried.

- $\beta 2 : 0.999$

  20 values from 0.699 to 0.999 were tried.

- $\epsilon : 10^{-7}$

- *Batch size* : 500 101 values, from 1 to 1000, which are multiples of 10 and 1 were tried.

- *Epoch number* : 250 In total of 17 values, from 1 to 400, that are divisible by 25 except 1.

- *Dropout rate* : 0.5

  4 values, from 0.5 to 0.9 were tried.

Third model we built is a decision tree. There are different optimization techniques such as information gain and Gini index and they do not take any parameters. We used a model that uses Gini index for optimization. Gini values and the real model are shown in Figure 3.8.

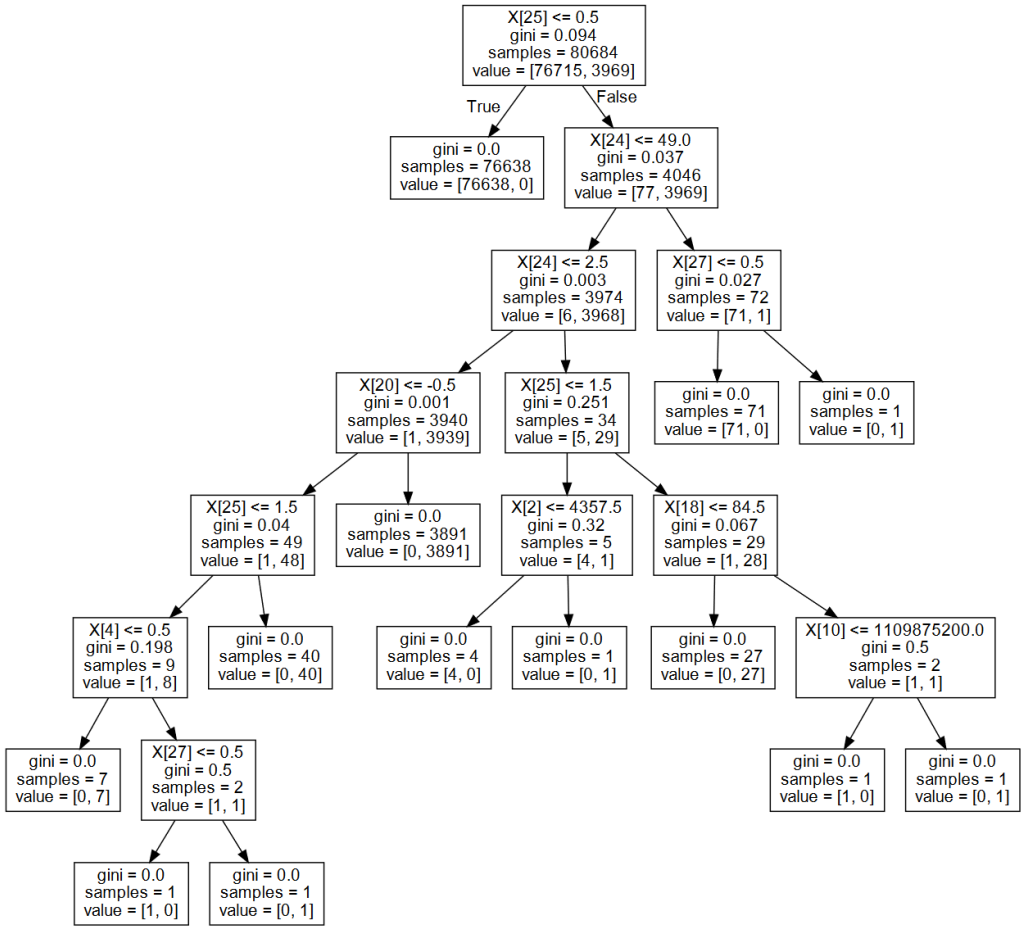


Figure 3.8: Actual Decision Tree model with depth of 8.

In order to aggregate the results of these three models, we used averaging. Computation of the output can be seen in Figure 3.9. Therefore, to have the final result, average of 3 results between 0 and 1 is computed. The output of the ensemble model is written into a basic text file in the format of "MSISDN %SuspicionRate".

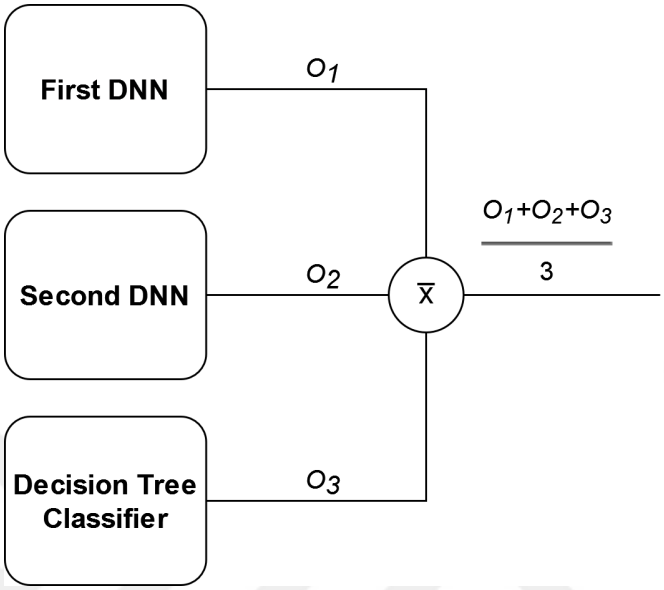


Figure 3.9: Ensemble model, output is the average of output of three individual models.

3.4 Model's Integration into Hydra Telecom

For the integration of our ensemble learning model into the visualization tool Hydra Telecom, we added a query option titled "Show potential fraudulent MSISDNs" to the "Custom Queries" title as shown in Figure 3.10. When this query is executed, text file containing the output of the learning model is read by Hydra Telecom, information of the suspicious MSISDNs are retrieved from its inner database and based on the parameters given to the query, the fraudsters are listed and/or visualized. Visualization is done by visualizing MSISDNs, people, IMEIs and dealers as nodes and calls and sms as edges.

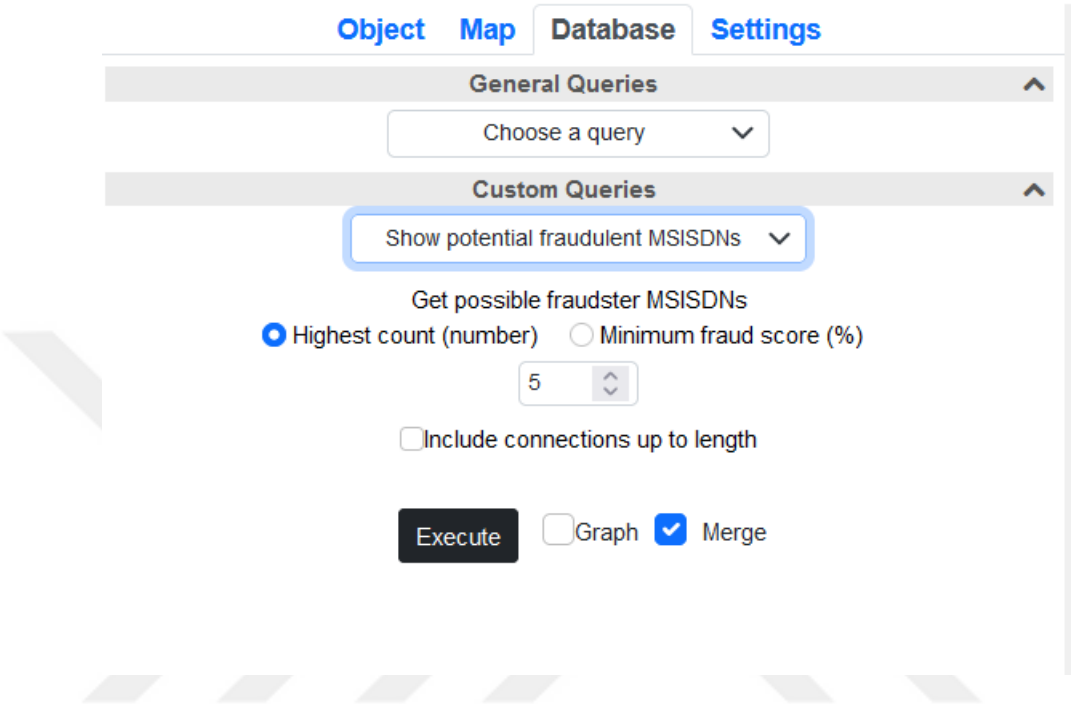


Figure 3.10: Tab of the "Show potential fraudulent MSISDNs" query.

The query has two options:

- First # with highest percent of suspicion (Highest count) and
- Users with at least %# suspicion rate (Minimum fraud score).

"#" symbol is the only numerical parameter given to the query. If the "Graph" choice is selected, the corresponding graph is visually presented as well as the numbers getting listed as a table as seen in Figure 3.11.

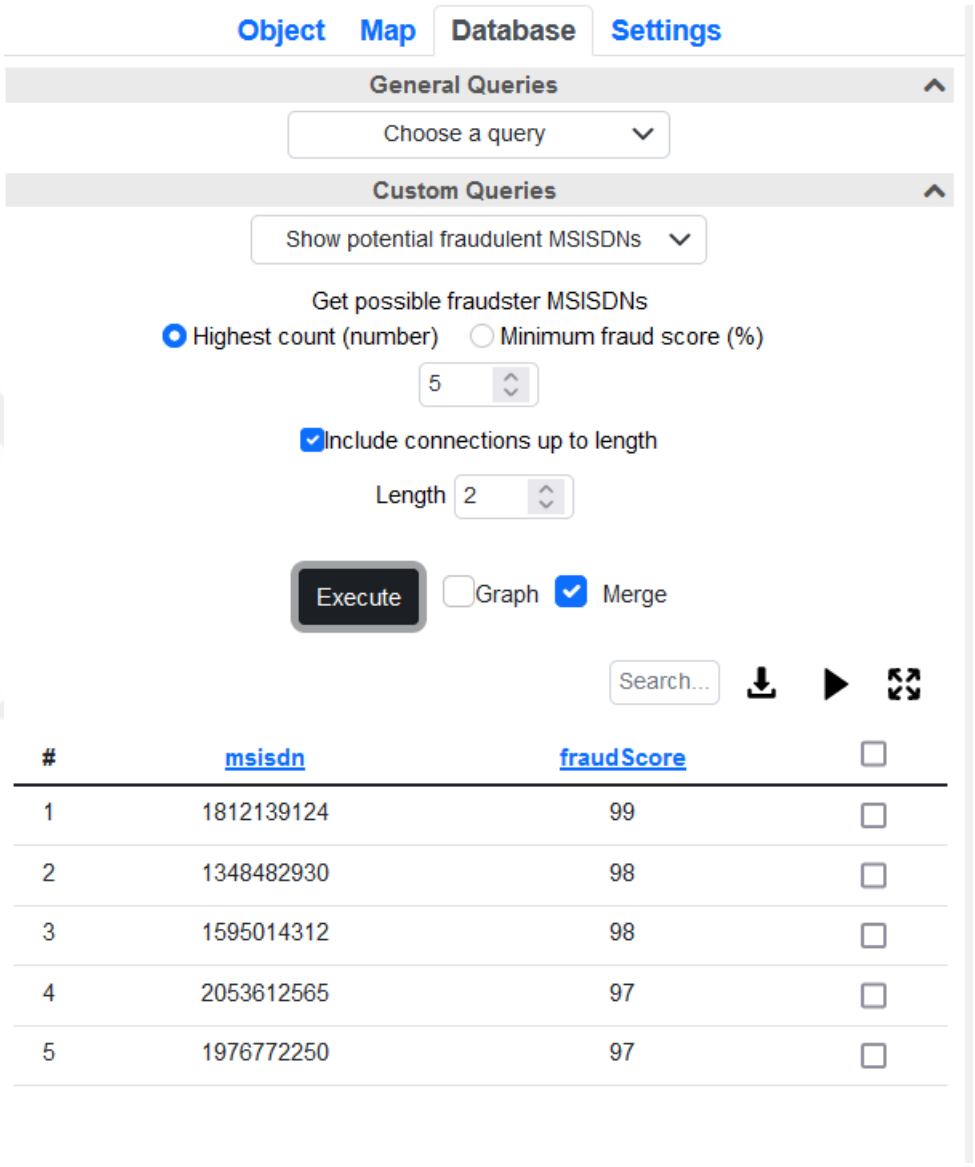


Figure 3.11: Tab of the "Show potential fraudulent MSISDNs" query.

Furthermore, we added an option titled "Include connections up to length". This option executes another query which can be explained as follows:

- When it is selected, another input box titled "Length" appears.
- "Length" value signifies the hop number, which is the maximum number of connections such as calls or sms between two fraud MSISDNs.

- Query returns the graph which contains the fraud MSISDNs based on the aforementioned parameters and the normal or fraud MSISDNs that stay in between any related pair of two fraud MSISDNs where their connection is at most up to the input "Length".

This query was adapted from and based on a query named "graph of interest" in previous work on biology network querying [24].

Chapter 4

Evaluation

We evaluated the quality of the training of our deep learning models using accuracy, precision and recall scores. However, since the class imbalance is high, 1:20, accuracy scores of all models were above 98%. Therefore, except for measuring the usefulness of our loss function, we also utilized the precision and recall scores to determine the hyper-parameters. Accuracy, recall and precision plots of validation and training sets for deep learning models can be seen in Figures 4.1-4.6.

Figure 4.1: Plot of first model's training and validation set accuracy score with respect to epochs.

Figure 4.2: Plot of first model's training and validation set precision score with respect to epochs.

Figure 4.3: Plot of first model's training and validation set recall score with respect to epochs.

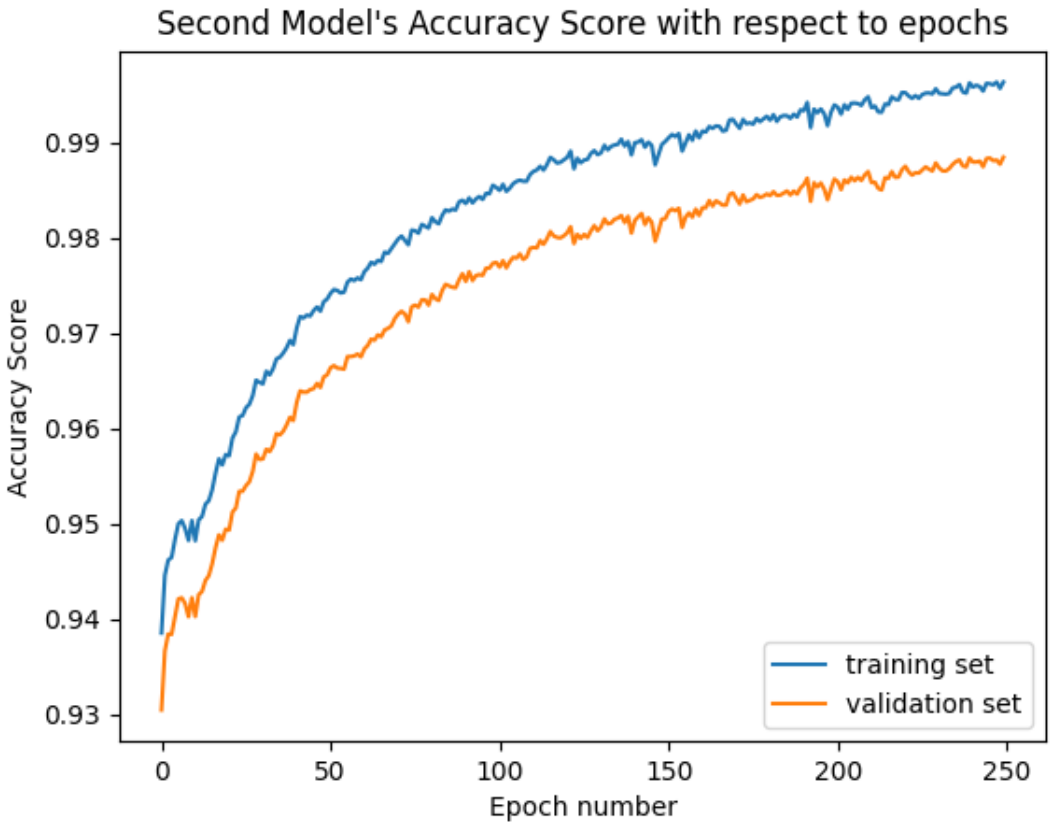


Figure 4.4: Plot of second model's training and validation set accuracy score with respect to epochs.

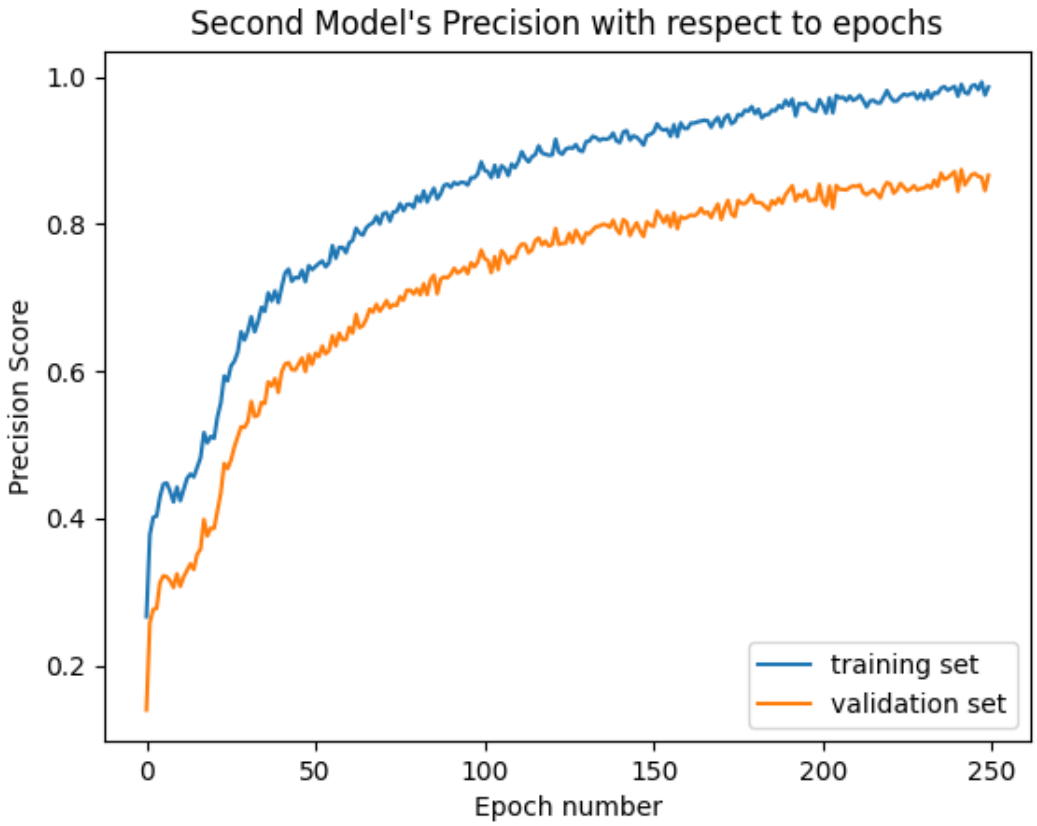


Figure 4.5: Plot of second model's training and validation set precision score with respect to epochs.

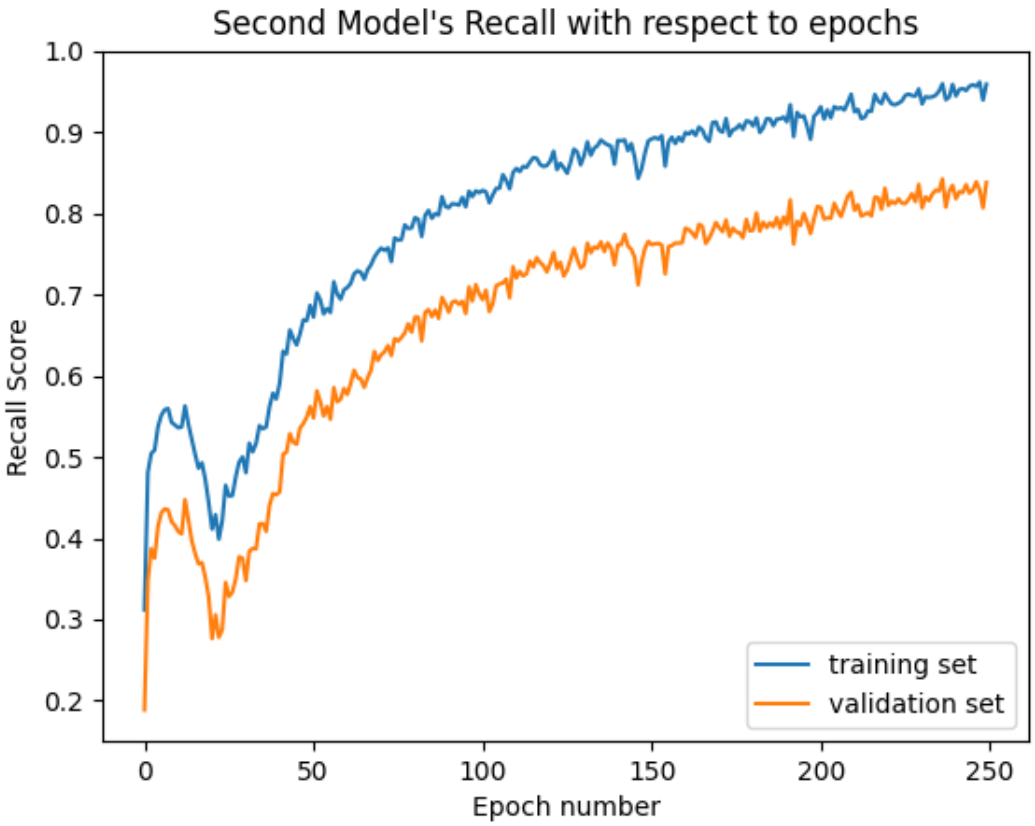


Figure 4.6: Plot of second model's training and validation set recall score with respect to epochs.

In order to measure the success of our model, we also used other machine learning methods with our data. These methods are as follows:

- A Graph Convolutional Network model (GCN) [25],
- A Relational Graph Convolutional Network (RGCN) [26] and
- A Support Vector Machine (SVM) [27] model.

Since GCN and RGCN are graph convolutional models, we used raw informations of nodes and edges as feature vectors. In SVM model we used the same features with the ones we used in our base model. Optimization methods such

as hyper-parameter tuning were done in each method. Then, we computed the accuracy, precision, recall and F1 score of each approach. We also measured the run time of each model from preprocessing stage to the results stage. Models' measures are shown in the Table 4.1.

Models	Accuracy	Precision	Recall	F1 Score	Run time
GCN	0.89	0.10	0.30	0.15	6 hours
RGCN	0.97	0.56	0.60	0.58	6 hours
SVM	0.98	1.0	0.56	0.72	2.5 hours
Our Ensemble Model	0.98	0.81	0.83	0.82	2.5 hours

Table 4.1: Comparison of different methods we applied on our data.

All these three models we compared our model with have different inefficiencies because of the nature of the problem. SVM produces closer results; however, it is biased towards normal users. Graph neural network (GNN) based methods have a different problem. As voice phishers are isolated in the telecom network, they do not reflect the local behaviour. Therefore, their feature vectors are diluted because of the message aggregation from their neighbourhoods. There are different GNN approaches that address this issue such as weighted message passing; however, manually featurizing some of the graph properties, i.e. the degree of a node into total number of calls, can be as equally or even more effective. Furthermore, building a graph from the raw data is a time-wise heavy process, it is less efficient for industrial applications unless the data is stored as graphs.

There are similar works which address Telecommunication Frauds. Although the datasets we use are different and comparing our results cannot provide a very good ranking, it can provide some insight. In the work "ConvNets for Fraud Detection analysis" [23], authors suggest the use of visualized features, which are similar to ours, with a Convolutional Neural Network (CNN). They generated 18000 images of the behavior or profile of 300 users in a 60-day period. Then, they used a Deep CNN for fraud prediction. Their accuracy score is 82% and they do not report their precision or recall values, or a confusion matrix. Another work titled "A Novel Method for Detecting Telecom Fraud User" [28] suggests

a method in which suspicious users are initially found by sms analysis. Fraud detection based on features extracted from CDRs using an SVM is applied on these initial results. In this work, accuracy, recall and F1 score are reported as 93.56%, 89.22% and 91.02%, respectively. Their measures except accuracy is higher than ours; however, they use approximately 8000 users for training and testing and their class imbalance, whose ratio is 1:3, is much lower than ours, which is unlikely in real scenarios.

Chapter 5

Conclusion

In this work, we present a combination of visual analysis and ensemble machine learning-based solution to detect Voice Phishing. Our ensemble model consists of two deep neural networks and a decision tree. We propose a set of features, which are claimed to be useful for Voice Phishing detection. To select these features, we reviewed the literature and made interviews with domain experts. To validate their quality, we analyzed graphical representations and statistics. Comparing our model's accuracy, precision and recall values (0.98, 0.81 and 0.83, respectively) to other machine learning methods and other works in the literature show that the features and the model can be useful for future studies. Also, with the visual analysis tool integration, it can easily be deployed as part of real life, industrial applications.

5.1 Discussion

This work provides a deep learning model for Voice Phishing detection with the set of features that are specifically engineered for this type of fraud. There are methods that provide higher precision and recall values; however, their success is either mainly due to the use of contents of calls and text messages [29] with

approaches such as Natural Language Processing or their datasets are composed of limited numbers of carefully picked instances [28]. We have a high number of instances which are only categorized and labeled. Also, the data used in our work does not contain contents of calls or text messages. The problem with content analyzing can be explained by two main reasons. The first one is that the size of the data and time of processing of the contents make the real applications infeasible. The second one is the privacy issues caused by such analysis. For individual mobile applications, legality can be ensured with terms of service agreements, whereas for broader applications laws may cause disputes. The ethical side of the problem is also debatable. Nonetheless, if these problems can be overcome, content analysis methods used besides our framework may give better results.

5.2 Future Work

Telecommunication networks are very suitable for representing and storing as graphs. Therefore, to profile and analyze user behaviour by analyzing the graphs of telecommunication networks can be expected to yield good results. Graph Neural Networks is a novel method that can be used with graphs; however, it has limitations. For instance, as we evaluated, in classification cases that aim to detect isolated nodes, even the current GNN models built for such use are generally less satisfactory than the less sophisticated Multilayer Perceptrons with appropriate feature selection. For that reason, our future work is to aim to utilize the representative ability of graph data type in machine learning applications which have similar characteristics to our case.

Bibliography

[1] A. Wilson, "An artificial neuron." https://towardsdatascience.com/inroduction-to-neural-networks-in-python-7e0b422e6c24.

[2] S. Jadon, "Introduction to different activation functions for deep learning," *Medium, Augmenting Humanity*, vol. 16, 2018.

[3] U. Dogrusoz, M. E. Belviranli, and A. Dilek, "CiSE: A circular spring embedder layout algorithm," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 6, pp. 953–966, 2012.

[4] M. Zaman, "An orthogonal layout algorithm for small compound graphs," 2021.

[5] Hiya, "State of the call." https://www.hiya.com/state-of-the-call, 2022.

[6] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital signal processing*, vol. 73, pp. 1–15, 2018.

[7] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody, and S. D. Brown, "An introduction to decision tree modeling," *Journal of Chemometrics: A Journal of the Chemometrics Society*, vol. 18, no. 6, pp. 275–285, 2004.

[8] Y. S. Canbaz, "Visuall: a quickly customizable library for jumpstarting visual graph analysis components," 2021.

[9] S. E. Griffin and C. C. Rackley, "Vishing," in *Proceedings of the 5th annual conference on Information security curriculum development*, pp. 33–35, 2008.

[10] Interpol, "Social engineering scams." https://www.interpol.int/Crimes/Financial-crime/Social-engineering-scams, 2022.

[11] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, "1 - an overview of machine learning," in *Machine Learning* (R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, eds.), pp. 3–23, San Francisco (CA): Morgan Kaufmann, 1983.

[12] M. Aparicio and C. J. Costa, "Data visualization," *Communication design quarterly review*, vol. 3, no. 1, pp. 7–11, 2015.

[13] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, "Algorithms for drawing graphs: an annotated bibliography," *Computational Geometry*, vol. 4, no. 5, pp. 235–282, 1994.

[14] S. G. Kobourov, "Spring embedders and force directed graph drawing algorithms," *arXiv preprint arXiv:1201.3011*, 2012.

[15] U. Doğrusöz, B. Madden, and P. Madden, "Circular layout in the graph layout toolkit," in *International Symposium on Graph Drawing*, pp. 92–100, Springer, 1996.

[16] U. Dogrusoz, Q. Feng, B. Madden, M. Doorley, and A. Frick, "Graph visualization toolkits," *IEEE Computer Graphics and Applications*, vol. 34, no. 2, pp. 30–37, 2002.

[17] Y. Koren, "On spectral graph drawing," in *International Computing and Combinatorics Conference*, pp. 496–508, Springer, 2003.

[18] K. Sugiyama, S. Tagawa, and M. Toda, "Methods for visual understanding of hierarchical system structures," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 2, pp. 109–125, 1981.

[19] K. C. Cox, S. G. Eick, G. J. Wills, and R. J. Brachman, "Brief application description; visual data mining: Recognizing telephone calling fraud," *Data mining and knowledge discovery*, vol. 1, no. 2, pp. 225–231, 1997.

[20] R. A. Becker, C. Volinsky, and A. R. Wilks, "Fraud detection in telecommunications: History and lessons learned," *Technometrics*, vol. 52, no. 1, pp. 20–33, 2010.

[21] C. S. Hilas, P. A. Mastorocostas, and I. T. Rekanos, "Clustering of telecommunications user profiles for fraud detection and security enhancement in large corporate networks: a case study," *Applied Mathematics & Information Sciences*, vol. 9, no. 4, pp. 1709–1718, 2015.

[22] C. S. Hilas and J. N. Sahalos, "An application of decision trees for rule extraction towards telecommunications fraud detection," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pp. 1112–1121, Springer, 2007.

[23] A. Chouiekh and E. H. I. EL Haj, "Convnets for fraud detection analysis," *Procedia Computer Science*, vol. 127, pp. 133–138, 2018. Proceedings of the First International Conference on Intelligent Computing in Data Sciences, ICDS2017.

[24] U. Dogrusoz, A. Cetintas, E. Demir, and O. Babur, "Algorithms for effective querying of compound graph-based pathway databases," *BMC bioinformatics*, vol. 10, no. 1, pp. 1–16, 2009.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[26] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European semantic web conference*, pp. 593–607, Springer, 2018.

[27] C.-C. Chang, "A library for support vector machines," *http://www.csie.ntu.edu.tw/˜cjlin/libsvm*, 2001.

[28] R. Li, Y. Zhang, Y. Tuo, and P. Chang, "A novel method for detecting telecom fraud user," in *2018 3rd International Conference on Information Systems Engineering (ICISE)*, pp. 46–50, IEEE, 2018.

[29] Q. Zhao, K. Chen, T. Li, Y. Yang, and X. Wang, "Detecting telecommunication fraud by understanding the contents of a call," *Cybersecurity*, vol. 1, 12 2018.

Appendix A

Data

Samples from the MSISDN information file:

5321031863452 510788912 1031863452 1021793296 0 239 28-DEC-21 28-DEC-21 12 06-JAN-22 3 93 2 2 X 6 X X X X X X 01-JAN-99 Yeni Tesis- Kampanya 0 N a a X

5321031860461 510787427 1031860461 1021790533 0 201 28-DEC-21 28-DEC-21 12 24-FEB-22 3 93 2 2 X 27 X X X X X X 01-JAN-99 Yeni Tesis 1 N a a X

5321033691064 511680602 1033691064 1023243864 0 201 15-MAR-22 15-MAR-22 12 05-APR-22 3 93 2 2 X 2 X X X X X X 01-JAN-99 Yeni Tesis 0 N a a X

5321031915503 510809353 1031915503 1021838938 0 201 30-DEC-21 30-DEC-21 12 05-APR-22 3 93 2 2 X 1 X X X X X X 01-JAN-99 Yeni Tesis 1 N a a X

5321033498464 511545359 1033498464 1023066581 0 201 05-MAR-22 05-MAR-22 12 10-MAR-22 3 93 2 2 X 38 X X X X X X 01-JAN-99 Yeni Tesis 0 N a a X

5321031472224 510586546 1031472224 1021430262 0 239 12-DEC-21 12-DEC-21 12 05-APR-22 3 93 2 2 X 6 X X X X X X 01-JAN-99 Yeni Tesis- Kampanya 0 N a a X

5321014241143 500012999 1014241143 1006039712 0 239 21-DEC-19 21-DEC-19 12 04-JAN-22 3 88 2 2 X 34 X X X X X X 01-JAN-99 Yeni Tesis- Kampanya 1 N a a X

5321023935298 506639304 1023935298 1014861409 0 239 14-MAR-21 14-MAR-21 12 21-JAN-22 3 93 2 2 X 6 X X X X X X 01-JAN-99 Yeni Tesis- Kampanya 1 N a a X

5321025327068 507427632 1025327068 1015974260 22 371 20-MAY-21 20-MAY-21 12 05-APR-22 3 93 2 2 X 27 X X X X X X 01-JAN-99 PortOutAvea 1 N a a X

Example from the CDR file:

5321024807867 905446447660 10 149 6453436459 X X 01/07/2022 09:02:40

5321031116874 903582185127 10 136 4246323319 X X 01/07/2022 10:57:07

5321024807867 905446447660 10 241 6453460518 X X 01/07/2022 11:12:24

5321022678913 908507249999 10 41 6153316593 X X 01/07/2022 11:54:28

5321031116874 903582182593 10 64 4246323319 X X 01/07/2022 12:14:44

5321024807867 905425348955401 10 63 6453436459 X X 01/07/2022 10:41:49

5321031116874 902163390083 10 17 4246323318 X X 01/07/2022 13:11:44

532992736593 532982645499 10 58 0249595158 X X 01/07/2022 12:51:06

5321031116874 902163390889 10 142 4246323319 X X 01/07/2022 13:13:24

5321031116874 903582184448 10 30 4246323319 X X 01/07/2022 10:18:12

5321031116874 903582184459 10 22 4246323319 X X 01/07/2022 10:40:35

5321031116874 903582185106 10 16 4336355351 X X 01/07/2022 10:42:59

5321024807867 905425348955401 10 439 6453436459 X X 01/07/2022 09:52:53

5321031116874 903582184505 10 13 4246323319 X X 01/07/2022 12:20:49

5321031116874 903582184513 10 1 4246323319 X X 01/07/2022 12:23:45

5321031116874 903582184514 10 3 4246323319 X X 01/07/2022 12:24:01

5321031116874 903423214900 10 18 4246323319 X X 01/07/2022 13:44:32

5321031116874 903423218600 10 3 4246323319 X X 01/07/2022 13:48:23

5321031116874 903582185764 10 19 4246323319 X X 01/07/2022 12:02:20

5321031116874 903582182591 10 3 4246323319 X X 01/07/2022 12:14:08

5321031116874 903582180209 10 27 4246323319 X X 01/07/2022 11:36:24

5321031116874 902163395474 10 378 4246323319 X X 01/07/2022 12:49:48

5321031116874 902163390892 10 287 4246323319 X X 01/07/2022 13:16:34

5321024807867 905446447660 10 341 6453436459 X X 01/07/2022 12:43:36

532992736593 905055325646486 10 159 0249595158 X X 01/07/2022 12:36:01

5321031116874 902663739305 10 1 4246323319 X X 01/07/2022 14:22:23

Appendix B

Code

Python code for building the first model:

```
model = Sequential()
model.add(tf.keras.layers.Normalization(axis=-1))
model.add(Dense(64, input_dim=input_dim, activation="relu"))
model.add(Dense(64, activation="relu"))
model.add(Dense(32, activation="relu"))
model.add(Dense(1, activation="sigmoid"))

opt = keras.optimizers.Adam(learning_rate=0.00001)

model.compile(loss="binary_crossentropy", optimizer=opt,
  metrics=['acc', f1_m, precision_m, recall_m])
history = model.fit(x_train, y_train, epochs=175,
  batch_size=38, verbose=0, validation_data=(x_val, y_val))
```

Python code for building the second model:

```
model = Sequential()
model.add(tf.keras.layers.Normalization(axis=-1))
model.add(Dense(units=200,
```

```
        input_dim=input_dim,
        kernel_initializer='uniform',
        activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(units=200,
        kernel_initializer='uniform',
        activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(units=1,
        kernel_initializer='uniform',
        activation='sigmoid'))

opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(loss='binary_crossentropy', optimizer=opt,
  metrics=['acc', f1_m, precision_m, recall_m])

history = model.fit(x=x_train, y=y_train,
          validation_data=(x_val, y_val), epochs=250,
          batch_size=500, verbose=0)
```