T.C.

ALTINBAS UNIVERSITY
Electrical and Computer Engineering

DATA PREPROCESSING TECHNIQUES IN MACHINE LEARNING TO DETECT FRAUD CREDIT CARD TRANSACTIONS

IBRAHIM QAYS
ABDULJABBAR ALHAYALI

Supervisor
Prof. Dr. Oguz BAYAT

Istanbul, 2019

DATA PREPROCESSING TECHNIQUES IN MACHINE LEARNING TO DETECT FRAUD CREDIT CARD TRANSACTIONS

By

IBRAHIM QAYS ABDULJABBAR ALHAYALI

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2019

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

Academic Title Name SURNAME Co-Supervisor	Academic Title Name SURNAME Supervisor

Examining Committee Members (first name belongs to the chairperson of the jury and the second name belongs to supervisor)

Prof. Dr. Oguz BAYAT	School of Engineering and Natural Science, Altinbas University	
Prof. Dr. Osman Nuri UCAN	School of Engineering and Natural Science, Altinbas University	
Academic Title Name SURNAME	School of Engineering and Natural Science, Altinbas University	
Academic Title Name SURNAME	Faculty, University	
Asst. Prof. Dr. Tareq Abed MOHAMMED	Faculty, University	

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science

Asst . Prof. Dr. Çağatay AYDIN
Head of Department

Approval Date of Graduate School of Science and Engineering: ____/____/____

Prof. Dr. Oguz BAYAT
Director

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

IBRAHIM QAYS
ABDULJABBAR ALHAYALI

DEDICATION

This thesis is dedicated to my father, who taught me that the best kind of knowledge to have is that which is learned for its own sake. It is also dedicated to my mother, who taught me that even the largest task can be accomplished if it is done one step at a time.

ACKNOWLEDGEMENTS

In the name of Allah, the Beneficent and the Merciful. Praise and Gratitude be to Allah for giving me strength and guidance, so that this thesis can be finished accordingly.

I would like to thank my supervisors*:* Prof. Dr. OGUZ BAYAT and Asst. Prof. Dr. TAREQ ABED MOHAMMED. Please let me express my deep sense of gratitude and appreciation to both of you for the knowledge, guidance and unconditional support you have given me. I wish you all the best and further success and achievements in your life.

My deepest gratitude goes to my dearest parents, for their immense patience and unconditional support and encouragement throughout my life. sisters, cousins (especially Asst. Prof. Dr. SHAYMAA ALHAYALI). My friends and colleagues: thank you very much for what you have done for me. I thank you all for the companionship that has made this journey much easier. In fact, I do not need to list your names because I am sure that you know who you are.

I would like to thank all the Altinbas University, college of engineering, department of the electrical and computer.

Finally, I also thank the Iraqi Ministry of Higher Education and Scientific Research, the Iraqi Cultural Attaché in Ankara.

ABSTRACT

DATA PREPROCESSING TECHNIQUES IN MACHINE LEARNING TO DETECT FRAUD CREDIT CARD TRANSACTIONS

ALHAYALI, Ibrahim Qays Abduljabbar

M.Sc., Electrical and Computer Engineering, Altinbas University,

Supervisor: Prof. Dr. Oguz BAYAT

Co-Supervisor: Asst. Prof. Dr. Tareq Abed MOHAMMED

Date: July, 2019

Pages: 58

The data is increasing day by day in a large amount in this time of the world. Data preprocessing in Machine learning has big role to solve the data problem, Getting the information knowledge from large data sets now plays a big and significant role in different kind of businesses. Data analysis of a dataset can be used to extract the hidden pattern to produce useful information. Machine learning is powerful tool for predictive analysis which predict the direct variable without focusing on complex relationships in the dataset. Data preprocessing is the important factor of machine learning which removes the complexity of data set. With these techniques we build predictive models based on real life credit card transactions data which allow us to find the new transaction is fraudulent or non-fraudulent. The aim and mission of this thesis is to introduce the difficult steps in machine learning like Data preprocessing, Real life example will show the importance and impact of data preprocessing in the machine learning. Credit card fraud is growing each year almost 1 percent of fraud has calculated each year for all the credit card transactions

which caused a lot of money to the business owners. Different kind of Machine Learning algorithms are used to solve the credit card transaction fraud problem. Main challenge is the large data set which contains almost 99.4 percent of data was non-fraudulent. This was a big challenge for data preprocessing. Data preprocessing techniques like data cleaning, normalization, feature extraction and selection are used to prepare the data for machine learning algorithms such as neural networks, decision trees, random forest, under sampling etc. In the result of some algorithms the accuracy of our model increases to more than 90 percent. As we introduce the best algorithm for each step of data preprocessing to get the high performance of the data set in proposed model.

Keywords: Machine Learning, Data Preprocessing, Deep Learning, Fraud Detection, Data Prepearation

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

LIST OF ABBREVIATIONS

ML	:	Machine Learning
KNN	:	K-Nearest Neighbors
DL	:	Deep Learning
SVM	:	Support Vector Machine
MP	:	Multilayer Perceptron
NN	:	Neural Network
TP	:	True Positive
TN	:	Ture Negative
FP	:	False Positive
FN	:	False Negative

1. INTRODUCTION

In this chapter I will give a short overview of this topic, In first section we will focus on the previous work done on the data preprocessing in machine learning and how it effects real data. This chapter will finish the general outline of thesis.

Last couple of decades has experience the revolution in technology and data engineering. Which made easy all the work of day to day operations in almost all the departments, we will be focusing on the decision-making part of the business and will help to solve the critical problems. All the business is producing large amount of raw data in daily basis via social media and from any other platforms and that's why it is very hard to observe this data and make good decision. People are investing in the data to use it in the decision-making process and which later turns into operational applications. Machine learning can solve complex problem which extract the useful points and patterns from the raw datasets and build a relation between variables to create useful knowledge or information.

Machine learning becoming more famous in research work and daily life applications and software's. The most important areas where machine learning is more critically used in pass decades are weather prediction, ecommerce business, face detection etc. Machine learning is also the most interesting area for researchers and most the Universities and research groups are investing in machine learning and data preprocessing. As from the name machine learning, it is the learning from experience, the process contains observing and analyzing the data and make future predictions by learning from the past. This research work will be more focus on the data preprocessing techniques in machine learning like supervised learning, Supervised learning models expect input and will predict a productive output.

This research work will solve a problem of credit card fraud detection. The credit card transactions data is available for research which we will use it in our models. Fraud in credit card credit card transactions are increasing day by day, To detect the fraud transactions for the companies will impact the profitability of these companies. The process needs to be deal with large amount of data that maybe incomplete or inaccurate, Data preprocessing techniques will take care of the incomplete or inaccurate data.

This process will start from data preparation where we will know whether data is clean , reliable, sufficient for this process. Data cleaning will deal with the missing data, inaccurate values, or other inconsistencies in the dataset. Data selection will deal with two issues like the relative importance of each attribute and the other is how to categorize all the attributes in the data. This thesis classifies the attributes in to continues and categorical data categories and then develop the algorithm to solve the problem based on attributes. Data preprocessing is the step when the clean data is enhanced sometime this enhancement involves creating new data from one or more than one fields in dataset, sometime replacing several fields with single field which contains more information or sometime the data needs to be transformed into a form which is acceptable as input for a specific machine learning algorithm. Data representation explains is to present the variable values in such away that the neural network can discern the differences between values and tell the relative magnitude of differences if that information is available. In model selection the back propagation neural network model is chosen to predict the target data, because the nature of prediction task is supervised. To fill missing data fields, self-organized-map model is selected to cluster all the data into subgroups where data have the similar characteristics. Separate Training and testing datasets. For most supervised neural network models, network begins with the training process. To obtain the best results, the critical neural network parameters should be set properly, they are learning rate, momentum, error tolerance, activation function and learning rule, neighborhood and number of cpochs. The criteria for the measure of the success is different in classification and clustering, the measure of success in classification problem is the accuracy of the classifier, usually defined as the percentage of correct classifications.

In methodology section of this thesis will implement and improve the previous models and algorithm of data preprocessing with detection of fraud transactions example. In this example an ecommerce business sells books which already sold thousands of books in last book. Now we build a fraud detection system which will show us the new transaction is fraudulent or non-fraudulent. Data set is explained in this chapter in which our first big problem is the vast majority users will not be fraudulent which made it harder to detect the underlying patterns in the information available which will cause our dataset highly unbalanced. Where we must use some sampling techniques and use different metrices to balance the dataset.

The dataset contains over three hundred thousand transactions all with their corresponding labels, Each one of the row in the dataset represents a user transactions with all the credit and user transaction available. Our goal is to be the classifier that given a new transaction can tell if its fraudulent or not with corresponding confidence. For data visualization jupyten nootbook will be used to all implementation and loading the dataset. Some important libraries will be used such as numpy, pandas, keran and random seed. After loading dataset and libraries the next step will be data preprocessing which is an important step in our case, because our data is unbalanced.

Machine learning techniques is the science of where computers are able to work like humans, like humans to extract pattern in data and take some action on the findings. Deep learning is the machine learning techniques which can be supervised, semi supervised and unsupervised. Here we will use supervised learning techniques which means we will need to teach our network with some input data and will expect an output as well. In this case the input will be a new credit card transaction and the output will be the information which will say whether this transaction is fraudulent or non-fraudulent.

The dataset will be split into two sub dataset which will called training and testing dataset. To start training our model we will need to call the compile method and will use accuracy to measure our results and then we will try to fit the problem, We will also face some overfitting problem in our model which we will remove with some algorithms.

The confusion matrix show us the performance of our model. In our case we will need to know how many fraudulent transactions we put into non-fraudulent transactions. If we only measure the percentage of their correct labels we will get a really high value as most of our transactions are known for non-fraudulent. If our model always predicts non fraudulent we will just get the majority of them right.

Decision trees is machine learning classification algorithm which will be used to test our both datasets (test dataset and train dataset). Under sampling techniques will be used to stabilize the dataset by locating our fraudulent transactions indexes in our dataset.

This thesis deals with automatic methods based on data and based on machine learning techniques. Practitioners must decide the function to use, strategy (e.g. supervised or unsupervised), the algorithm (e.g. decision trees, neural network, support vector machine),

frequency of update of the model (once a year, every month). or new data is still available) etc. We hope that the reader will better understand the design and implementation of an effective data-driven fraud detection solution at the end of the thesis.

2. LITERATURE REVIEW

2.1 MACHINE LEARNING

Machine learning is trending in scientific research work and plays an important role in the software and other applications in our daily life. Most of the time machine learning is used for weather predictions, fraud detection in any kind of bank or credit transactions, face detection software's, product recommendation like Amazon do in their software to show the related products to you on your search history or your browser cookies etc. Machine learning is the study of learning the knowledge from experience, this process contains observing the problem and developing a hypothesis which help the machine learning to predict the future behavior of the system. In our case as we are using the machine learning to solve our data problem in this case we will be providing data to the machine learning model which will extracts the knowledge from raw data set to the useful information. Machine learning is closely related to the pattern recognition, data mining and statistics at the same time it is related to the computer science where it focusses on those algorithms which are involve in the extraction of information from data. The main idea of machine learning is to create algorithms which learn the patterns automatically from the data.

This thesis is about data preprocessing and machine learning techniques like supervised learning, where system will expect and input and will predict a productive output which is call supervised learning. There is another term called unsupervised learning in which the response or the output variable are not available. While supervised learning assumes the availability of an input and an output where the machine learning models can compare the predicted output to the original output and can check the accuracy and precision. Data preprocessing will contain the data cleaning, dealing with missing data, dealing with outliers, data transformation, centering and scaling, data reduction and feature extraction, and in supervise learning statistically we will be focusing on formalization, classification etc.

We use some real-life problems and try to solve it with data preprocessing techniques and supervise learning, Fraud detection problem is the more challenging one where the observations of the model will be the transactions which can be either non-fraudulent or fraudulent.

2.2 DATA PREPROCESSING

The real-world data is imperfect and insufficient which means data have a lot of errors missing values etc. Data preprocessing is the important step of machine learning which helps to convert the raw data into useful information. Some of the modeling techniques are very sensitive that's why it makes data preprocessing more important, here we will focus on data cleaning, data transformation and data reduction which are all the processes of data preprocessing.

2.3 DATA CLEANSING

2.3.1 Dealing with Missing Data

In the real world it is very common that some data is missing, and sometimes these missing data can affect the results of the model. We must be very careful and understand why some data is missing. In some cases, the missing data becomes random and refers to a part of the predictive models, but the result is not related to missing values or data that is still called population. However, if some models of the forecast model do not contain any data, our results are unreliable. Some of the proposed procedures for cleaning up data relate to missing data. We will focus on two of them. The simplest solution that is proposed is to remove the missing values immediately. We can use this technique if the record is very large and the missing data is not related to the predictive model and the missing data is random. If we delete the missing data, this does not have much influence on the forecast results. If one of them is not there then using removing directly missing data can cause some problems in the results. The next solutions proposed to the missing data is to fill the missing values related to the given data, we can use average of the missing values and fill the missing values and other technique is to use the decision trees to find the missing vales.

2.3.2 Dealing with Outliers

The most efficient way to remove the outliers are to visualize the data then we can find out the values which are outliers and significantly valid or not, to remove the outliers you must have the valid reason, either the values are totally different from main stream data or the values are wrong. Instead of deleting the data, there is another way to minimize the effects of outliers.

2.4 DATA TRANSFORMATION

Data Centering and Data scaling of data are the simplest technique of data transformation. This is the most basic part of the pre-processing data in the analysis. Some machine learning and data mining techniques, such as Euclidean Distance, require data to be scaled and centered before being entered into the model. The Euclidean distance is the normal straight line between two points of Euclidean space. This helps in data transformation to improve the interpretability of parameter estimation for interaction in the model. Data scaling scale the data and each prediction value are arranged according to the standard deviation by which scale data has unit deviation. Data scaling will affect nothing if data has outliers so before data scaling it is important to remove the outliers from the data, Data Centering subtracts or suppresses the average value of the predictors so that each predictor or dataset has the mean of zero.

Many statistical models assume that all the data are examine in normal distribution but data can be positively skewed or negatively skewed. To normalize the data first step should be to resolve the skewness with applying transformation. As we know there are more data transformation techniques which help is to resolve the skew, for example we can use the inverse transformation, square root or log. In current example and in fig 1 you can see how the log transformation solve the skew problem.

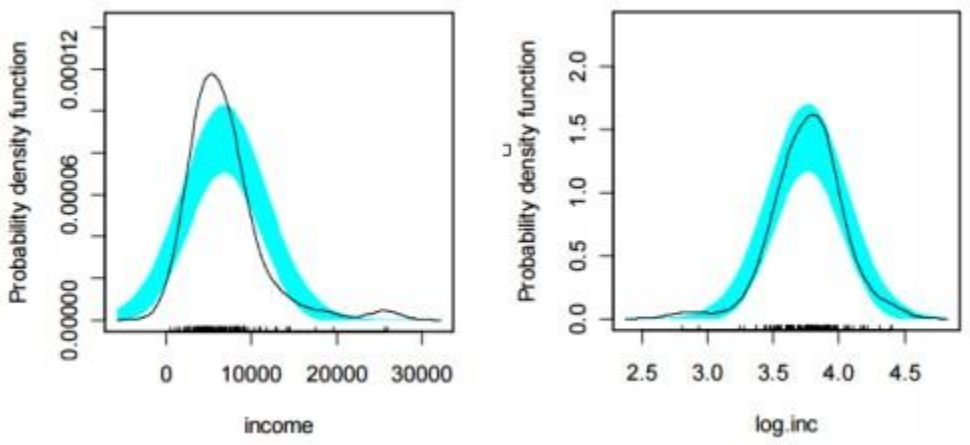

Figure 2.1: On left Hand side: a positive-skew data distribution. On right hand side: the same data after a log transformation.

Box and Cox (1964) propose a set of transformations indexed by a parameter λ that can empirically identify an appropriate transformation. Eq 2.1

$$x^* = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(x) & \text{if } \lambda = 0 \end{cases} \tag{2.1}$$

Equation (2.1) can identify various transformations, such as log transformation, square root (λ=0.5), square (λ=2), inverse (λ=-1), and other in-between transformations.

2.5 NEURAL NETWORK

Bishop et al (1995) proposed a network, who then called with the name of the neural network. The neural network is a very powerful network that is inspired by the human nervous system, such as the brain and the way the brain processes information. The neural network is a nonlinear regression technique. The neural network consists of a number of small and interconnected

processing elements called hidden layers or hidden units, also called neurons. The linear combination is usually transformed by a non-linear function such as a sigmoidal function: Eq(2.2),(2.3)

$$h_k(x) = g\left(\beta_{0k} + \sum_{i=1}^{P} x_j \beta_{jk}\right) \tag{2.2}$$

$$\text{Where } g(u) = \frac{1}{1+e^{-u}} \tag{2.3}$$

After the hidden layers are determined, another linear combination is applied to connect the hidden layers to the results Eq(2 . 4).

$$f(x) = \gamma_0 + \sum_{k=1}^{H} \gamma_k h_k \tag{2.4}$$

For this kind of neural network models with N number of predictors, the total estimated parameters are H(N +1) + H +1. It is obvious that with the increase in number of N, the number of parameters will become high, which proves that the pre-processing data and removing irrelevant predictors is an important step to reduce computation time.

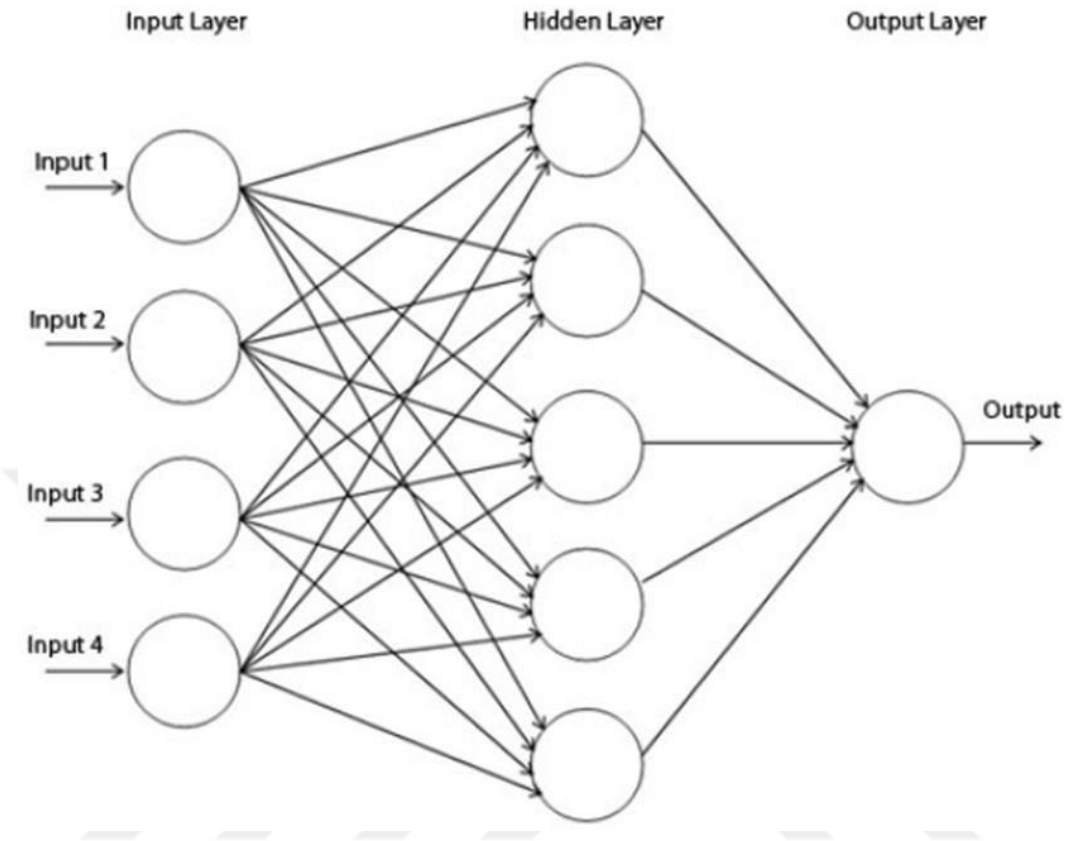

Figure 2.2: Feed forward neural network with a single hidden layer

The neural network parameters begin with random values and are then imported into some important algorithms, e.g. Gradient descent or Bayesian algorithms which are then applied to the sum of square residuals. The model of Neural network can not guarantee an authentic solution but it can give us optimal solution to the problem. The better way to create several neural networks and then combine and compare the results to get more useful prediction.

As we have been discuss that the Neural networks tune the parameters of model before provide it to the system, More often it gives us the negative impact on our model. So which proves again that data preprocessing is important to help build a reliable and useful model, which also speed up the computation problem of the task.

Overfitting refers to a model that models the learning data too well. Overfitting occurs when a model learns details and noise in the learning data as the performance of the model is adversely

affected by the new data. That is the random variation in the training data set which model save as the method or concept. To generalize the model these concepts must apply to the new data or for example to the testing data set, but in overfitting it does not apply these concepts to the new data which is the problem.

Over-fitting more fit for non-linear and non-parametric models which is more diverse to learn a specific target method. For example, decision trees are non-parametric algorithm of machine learning.

2.6 MULTILAYER PERCEPTRON

A multilayer perceptron is a neural network that connects multiple levels in a target graph. This means that the signal path through the nodes only runs in one direction. With the exception of the input nodes, each node has a non-linear activation function. An MLP uses backward propagation as a guided learning method. Since there are different layers of neurons, MLP is a deep learning method. A multilayer perceptron (MLP) is a feedforward artificial neural network that generates a set of outputs from a set of inputs. Fig (2.3) shows two layers perceptron.

MLP is often used to solve problems that require supervised work, as well as computational neuroscience research and parallel distributed processing. Applications include speech recognition, image recognition and automatic translation.

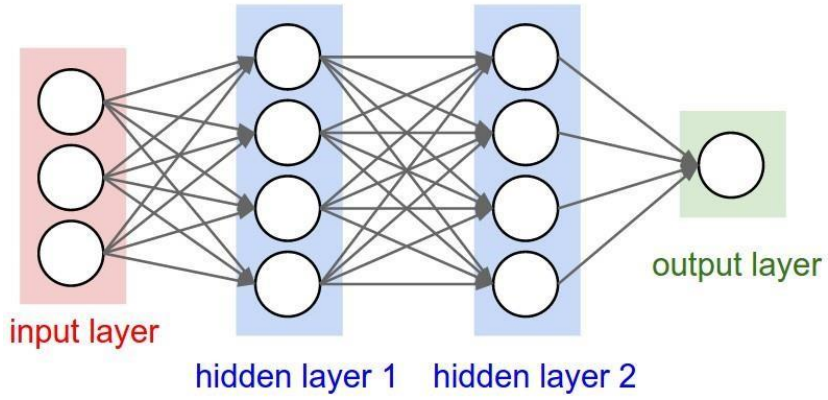

Figure 2.3: A diagram of multilayer perceptron with 2 hidden layers

2.7 SUPPORT VECTOR MACHINE

SVMs are a set of associated supervised learning methods used for classification and regression. They belong to a family of general linear classifiers. Support Vector Machine (SVM) is a classification and regression prediction tool that uses the theory of machine learning to optimize the accuracy of predictions and automatically avoid data over-adjustment. Support vector machines can be defined as systems using a hypothesis space or linear functions in a large function space formed with an optimization algorithm learning algorithm implementing a training distortion derived from the theory of statistical learning.

Support vector machines are really very effective in large spaces. It's even very efficient for datasets where the number of dimensions is greater than the number of samples. Other advantages of support vector machines include memory efficiency, speed, and overall accuracy over other classification methods such as the nearest k-nearest neighbours or deep neural networks. Of course, they are not always better than e.g. deep neural networks, but sometimes they are always better than deep neural networks.

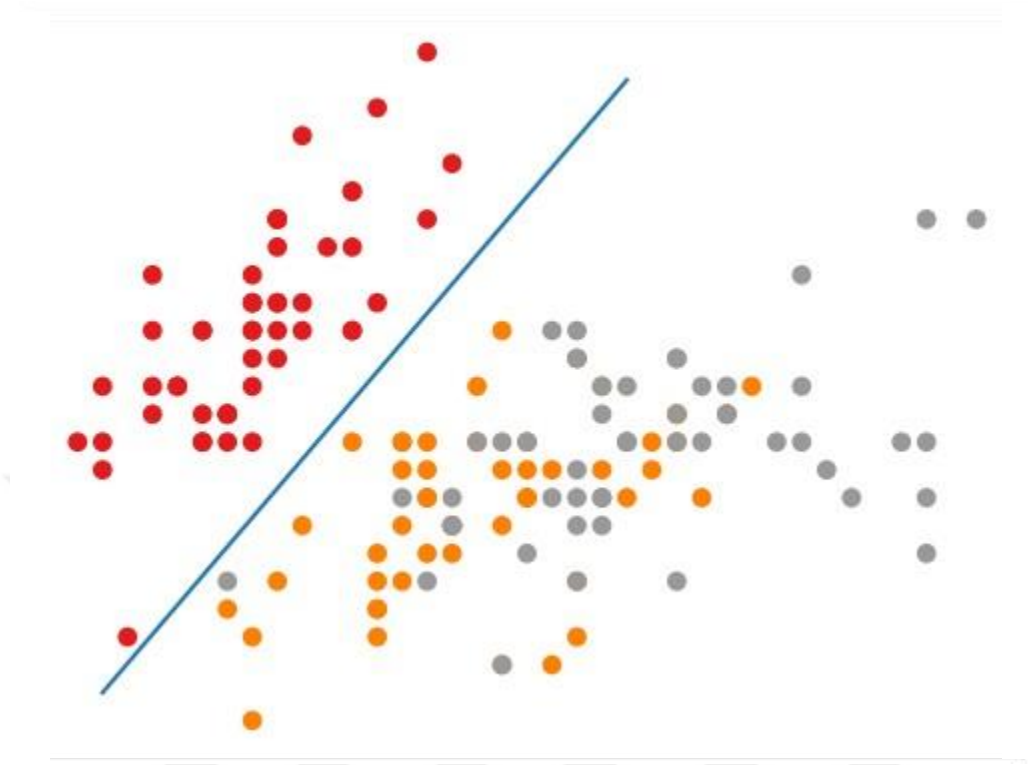

Figure 2.4: Support Vector Machine, Best Possible way to split data

2.8 K-NEAREST NEIGHBORS

K-Nearest Neighbors (KNN) algorithm is simple, easy-to-implement algorithm for supervised machine learning that can be used to solve both regression and classification problems. The KNN algorithms assumes that the similar things exist close to each other.

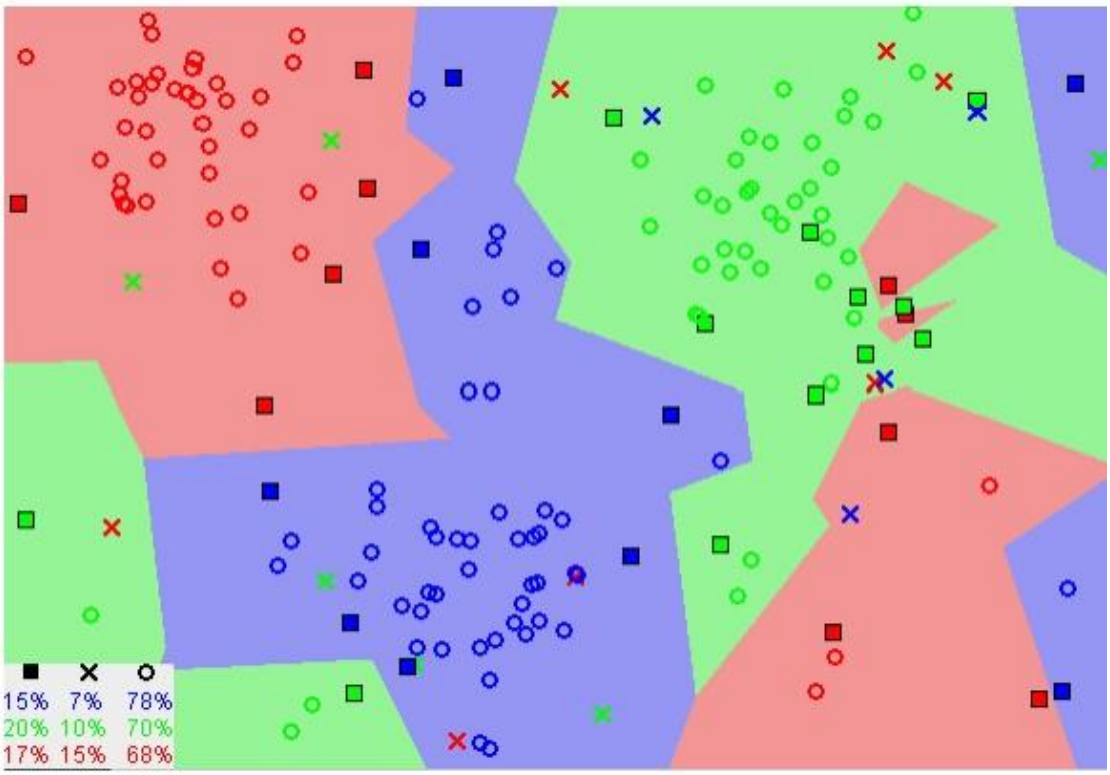

Figure 2.5: K Nearest Neighbors Algorithm**.**

In Figure 2.5 most of the time the same data points are close to each other and KNN use this assumption to be true for the algorithm to be effective. KNN has been used in statistical estimation and pattern recognition already in the beginning of 1970's as a non-parametric technique.

The k-means algorithm is an unsupervised clustering algorithm. It takes a bunch of unlabeled points and tries to group them into "k" number of clusters. It is unsupervised because the points have no external classification. The "k" in k-means denotes the number of clusters you want to have in the end. If k = 5, you will have 5 clusters on the data set.

The K-nearest neighbor algorithm is intuitive and simple and has a remarkable predictive capacity, especially when the answer lies on the local predictor structure. However, the calculation time is a noticeable problem. To predict a sample, the distances between the observations and all other observations must be calculated so that the calculation with K and the size of the data increase considerably.

2.9 OVER FITTING PROBLEM AND MODEL TUNING

2.9.1 TheProblem of Over Fitting

To select best good fit model, it must be in between underfitting and overfitting which is difficult to achieve this goal. To understand overfitting, we can look at the performance of algorithms of machine learning on training data set and test dataset. With this process algorithm will learns the error of the model on training dataset and on test dataset. If with train the error of the model on training data decrease and error of the model on test data increases which means the model is overfitting and learning the wrong details. The best spot in this process where we will this model is good for both data sets, when the error of the model start increasing. Overfitting is a problem because the evaluation of machine learning algorithm on training data is different from the evaluation of test data on ML algorithm.

2.9.2 Methods

There are two important techniques for evaluating machine learning algorithm to limit overfitting.

- You can manually search the list of features and decide which of the most important features you want to keep. A class of algorithms, called model selection algorithms, automatically selects the most relevant features. We will examine them in the next chapters. Although reducing the number of functions solves the problem of matching, it is not an ideal solution. The disadvantage is that you are throwing valuable information about the problem.
- Methods of Resampling are following the procedures in such a way that extract the sample from dataset and reshaping the model for each sample to learn more about the fitness of the model, this process is expensive because it same statistically procedures for N number of time for all the samples.

3. A SOLUTION TO CREDIT CARD FRAUD DETECTION PROBLEM

The task of approving customers for credit, assigning credit limits and detecting credit fraud is a labor-intensive and time-consuming process that has significant impact on the profitability of most companies [Trippi and Turban, 1996]. How to relieve labors from labor-intensive task and how to find efficient solutions to reduce time complexity become a very challenging and promising field. Furthermore, the process needs to deal with an extremely large amount of data that may be incomplete or inaccurate. How to handle such high-dimensional data is very crucial. An artificial neural network has demonstrated its usefulness in the analysis of such data sets with a distinctive new flavor, which deals with the diversity of input information without requiring that the information be restated in a standard form. It can be trained using customer data as the input vector and the actual decisions of the credit analyst as the desired output vector. This work uses neural network to deal with credit card record sets and develops innovative algorithms to fill missing data. The important steps are presented in the following sections.

3.1 DATA PREPARATION

This is the first step in data mining process. In most cases, the data used for a data mining operation has been just sitting around collecting dust [Bigus, 1996]. There are three important issues need to be concerned before mining these raw material:

- Are these data clean?
- Are these data reliable?
- Are these data sufficient?

3.2 DATA CLEANSING

It is often the true fact that not all operational transactions are correct. They might contain inaccurate values, missing data, or other inconsistencies in the data set. Several techniques are

being used to clean data. These include rule-based techniques for detecting inaccurate or inconsistent data, which evaluate each data item against metaknowledge (knowledge about the data) about the range of data expected in that field and constraints or relationship to other fields in the record [Simoudis et al., 1995]; Visualization can also be used to easily identify erroneous and out-of-range data. Another way is to use statistical information to replace missing or incorrect field values with neutral, valid values. Data cleansing in this work focuses on filling missing data rather than detecting 24 inaccurate data. Hence, We carefully develop algorithms to determine the default value for each attribute. The work first statistically analyzes each attribute in the data set and gets the probability distributions for all attributes. Then, designs algorithms to determine the default values for all attributes and replaces all missing data with these default values.

3.3 DATA SELECTION

Data selection concerns two central issues: one is how to determine the relative importance of each attribute. The other is how to categorize all attributes. A data set may have M attributes, which have the different contribution to the decision-making. How to determine the importance of attributes falls into two ways. Sometimes, experts can do this manually. However, this is a kind of case-by-case solution. Based on statistical analysis, another method can determine the attribute importance by comparing the similarity between target attributes and the uniform distribution functions. If the distribution of an attribute is very similar to the uniform distribution, this attribute has the less importance. The key point is if the distribution of an attribute is uniform, each value has the same probability to appear in a dataset, which makes it useless to predict missing data.

Since some attributes in a data set may have ambiguous value, attribute classification is another important issue in data selection. For instance, value "19" can be either interpreted as categorical data like department number, discrete number value like age or regarded as continuous numeric

data. This thesis classifies all attributes into two categories: continuous and categorical data and develops an algorithm to solve this problem based on attributes' output types. The dataset used in this thesis is from UCI Machine Learning dataset [52] library which is freely available for researchers.

3.4 DATA PREPROCESSING

Data preprocessing is the step when the clean data, which have been selected, is enhanced. Sometimes this enhancement involves generating new data items from one or more fields, sometimes it means replacing several fields with a single field that contains more information, and sometimes the data needs to be transformed into a form that is acceptable as input to a specific data mining algorithm, such as a neural network. There are several techniques listed below, which are mostly used in the data preprocessing phase:

- Computed attributes: A common requirement is to combine two or more attributes into a new attribute. This is usually in the form of a ratio of each combined value, the sum, the product or other values.
- Scaling: Another transformation involves the more general issues of scaling data feeding to the neural network. Normally, most neural network models adapt Sigmoid (Sig), Hyperbolic Tangent (TanH) or their variants as transfer functions. Equations are: (3.1),(3.2)

$$\text{Sigmoid: } f(z) = \frac{1}{1+e^{-z}} \tag{3.1}$$

$$\text{Hyperbolic Tangent: } f(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \tag{3.2}$$

The Sigmoid transfer function requires the input range [0,1] while the Hyperbolic Tangent requires the input range [-1, +1] to avoid "saturation" effect. Thus to scale data into the proper range requires another necessary data preparation step.

In this work, all neural network models such as Back-Propagation (BP), SelfOrganization-Map (SOM), etc. use the TanH transfer function; Thus, the original data are scaled into the [-1, +1] range.

Normalization: Vectors or tables containing numeric data can sometimes be treated as groups of numbers. In these cases, the vectors must be standardized as a group. There are many ways to deal with it. The most common vector normalization method is to add the squares or each element, take the square root of the sum, and then divide each element by its norm. Another way to standardize vector data is to simply add all the elements of the vector and divide each number by the sum. In this approach, the sum of the normalized elements is 1.0, and each has a value representing the percentage contribution they make. The third method is to divide each vector element by the maximum value of the array.

3.5 DATA REPRESENTATIONS

Neural networks explore lots of categorical data to do data mining. For categorical data the challenge of representation is to present these variable values in such a way that the network can discern the differences between values and tell the relative magnitude of the differences if that information is available. Various coded data types are used to represent these values. Binary code assigns each category a value from 1 to N and represents by a string of binary digits. Thermometer code is better used when the discrete values are related in some way, usually by increasing or decreasing values. For example, to represent morning, noon and night by 3 bits, noon, morning and night can be represented as [0.5, 1, 0.5], [1, 0.5, 0] and [0, 0.5, 1], respectively. This thermometer encoding scheme not only reveals the difference between categories, but also shows the relationship between each other: Noon is next to morning and night, morning next to noon but far from night etc

This work designs a different encoding scheme to represent categorical data. The detailed procedures are:

1. Analyze data types in each attribute and calculate each type's percentage.
2. Divide all data types into 3 parts, the left and right contain data types which percentage is the lowest and middle part contains the highest percentage data types

Subgroup all categories into 3 groups, which are encoded as [-0.355, -0.355, +0.355], [+0.355, -0.355, -0.355], and [-0.355, +0.355, -0.355], respectively. The 28 reason to choose value +/-0.355 is to make sure any pair has the Euclidian distance 1, in other words, to normalize the distance of each pair.

3.6 MODEL SELECTION

There are many different types of neural network models, which can be categorized by the basic learning paradigm or the approach they use. Supervised neural network is the most common training paradigm, which makes predictions or classifications for a given problem or case. At this point, the "desired" results act as a supervisor to indicate what the answer should be and how to update network itself to achieve this goal. The unsupervised model is often used for clustering and segmentation in data mining, where neural network doesn't have "standard answer", and the training goal is to group similar characteristic vectors together.

In this work, the Back-Propagation Neural Network model is chosen to predict the target data, because the nature of prediction task is supervised, which means there already have some sample cases made by experts in a corresponding field during learning phase. Back-Propagation models are just to learn how to deal with these cases, and simulate an expert to predict target data when a new case without "answer" feeding into a network.

To fill missing data, Self-Organization-Map (SOM) model is selected to cluster all records into some subgroups, where data have the similar characteristics. The key point is if the missing data belong to a subgroup, it is very possible for their values to fall into the value range determined by the subgroup's complete data records.

3.7 TRAINING AND TESTING THE MODEL

Once the data preparation is completed and the neural network model and architecture have been selected, the next step is to train the neural network that includes the following two steps:

1. Separate training and testing data sets: For most supervised neural network models, networks begin the training process with the connection weights initialized with small random values. The training control parameters are set and the training data patterns are presented to the neural network one after the other. As training progresses, the connection weights are adjusted, and the performance of the network is monitored. In order to evaluate the performance or to determine whether the target model succeed in data prediction or not, implementation needs to test data set, where all weights adjusted at training phase are fixed. This work splits the whole data set into testing and training parts and set the ratio at 1:3.
2. Set important neural network parameters: To obtain the best results, the critical neural network parameters should be set properly. They are:
    a. Learn rate: Control the step size for weight adjustments. Decrease over time for some types of neural network. Learn rate is very important because if value is too large, the error factor drop very fast but the side effect is easily falling into "unstable" state. The whole model could never meet the convergence criteria in worst case. On the contrary, whole network will gradually go into a "stable" state but take too much time. It is a trade-off. The proper value based on experience and practice. In the project BP and SOM both need to set this value.
    b. Momentum: Smooth the effects of weight adjustments over time. BP model need to set this value.
    c. Error tolerance: Specify how close the output value must be to the desired value before the error is considered to be zero. In BP, the probe using to determine error tolerance usually is Mean Square Root (RMS).
    d. Activation function and Learning Rule: Select the activation function (transfer function), which is used by the neural processing unit. Most common is the sigmoid or logistic activation function. As mentioned before, in order to get better performance, the project always extend input value range from [-1, +1], so for BP, the activation functions usually used are variants of TanH, learning rules are Delta-Bar-Delta, Ext DBD and Norm-Cum-Delta. For SOM, the functions are logistic or Euclidian.

e. Neighborhood: Defines the size or area of units surrounding the winner, which get their weights updated. Neighborhood decreases over time. SOM need to specify this value.
f. Number of epochs: determines the number of passes for networks that train for a fixed number of passes through the training data. BP and SOM both need to set this parameter.

3.8 MEASURES OFSUCCESS

Once a neural network model is selected and trained with data to check whether the model is well trained or not. The criteria for the measure of the success is different in the following steps:

- Classification: The measure of success in a classification problem is the accuracy of the classifiers, usually defined as the percentage of correct classifications. The algorithms to determine correctness are various. In this work, there are two evaluators. One determines the correctness based on whether the difference between a prediction value and desired value's RMS is lower than a specific threshold or not. The other tool is more sophisticated by using a classification matrix and a reverse matrix.

Clustering: Since clustering is an unsupervised usage, where there are no "standard" answers to verify a model. In most cases, the training regimen is determined simply by the number of times data is presented to a network, and how fast the learning rate and the neighborhood decay. A network will be trained for the certain number of epochs specified by a user and then stop.

3.9 SUMMARY

Data mining using Neural Network includes the following four steps:

1. Data Preparation: it involves data preprocessing, data representation, data selection, and data cleansing. Data selection is a kind of domain knowledge. The domain expert uses his knowledge of a problem and available data to determine attributes' importance and classification. Data cleansing in our work focuses on filling missing data. The only

purpose of filling missing data in data cleansing step is to gain records vectors, in which every element has a valid value and "erroneous" effects caused by missing values are minimized. In the data preprocessing phase, 32 this work scales and normalizes continuous data, and transforms categorical data into a numeric normalization form using a specific encoding scheme.

2. Neural Network Selection: Neural network models are selected based on required tasks. For data prediction purpose, the supervised architecture model BackPropagation Network is selected. To fill missing data, the unsupervised SelfOrganization-Map network is chosen.
3. Training and testing phase: lists the most important parameters, which directly affect the overall performance of neural network, and explains what these parameters are for.
4. Evaluation of results from neural network: this work not only uses a common tool such as RMS, but also designs two evaluators: an accuracy matrix and a reverse matrix.

4. METHODOLOGY

This chapter will explain Data preprocessing in machine and deep learning algorithms with real world data. Which will evaluate fraud data from credit card transaction data. It will be called credit card fraud detection. In this chapter both machine learning techniques will be used to solve our problem. Credit card fraud detection is important to any e-commerce business and these companies put a lot of investment to prevent fraud because even single fraud can cause them a lot of money.

This chapter will show how data preprocessing and machine learning techniques will help to solve this problem.

Ecommerce has changed everything and that's because it's important to minimize the fraud from ecommerce business. Ecommerce has given business man chance to increase sales but at the same time all the business man is exposed to hackers and other kind of fraud.

For this exercise we're supposed to do our own e-commerce store that sells books. We're still not Amazon but we have sold thousands of books during the last years. And today we're going to use our transaction history to build a fraud detection system. We use a publicly available data set for these with real credit card transactions that have been anonymized. One of the biggest problems of credit card fraud that action is that anyone can still get 15 or 16 digits of a credit card with a security code an expiration date so far. Our store has only use rules based systems to do fraud detection but we have seen that this is not enough. Rules can be as simple as, Is this the first purchase ever from this user which worth $500 or more complex ones such as credit card transaction is from China, The credit card was issued in the U.S. and they user in the U.K. It will be our job to tell about those transactions that come from our users that people who actually own the credit card that is being used to buy the goods that you are selling from those who have stolen credit card.

One of the biggest issues with this problem is that the vast majority of users will not be fraudulent. This will make it harder for us to detect the underlying patterns in the information available. This will cause our data set to be highly unbalanced. So we need to apply different

sampling techniques and use different metrics. Aside 99 percent accuracy one being that our solution is any good. It is estimated that only point 1 percent of online credit card transactions are fraudulent. But given the volume of transactions that occur every day that means a lot of money. The estimated cost of fraud in the U.S. last year alone is over nine billion dollars. And this number is increasing year over year.

4.1 MACHINE LEARNING VOCABULARY

Machine learning is used to solve this problem. The dataset that are going use has real anonymize credit card transactions that dataset contains over three hundred thousand transactions all with their corresponding labels. You can see that we have only numbers in the columns which has been anonymize.

Table 4. 1: Dataset with column names anonymized

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V21	V22	V23
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	...	-0.018307	0.277838	-0.110474
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.225775	-0.638672	0.101288
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	...	0.247998	0.771679	0.909412
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	...	-0.108300	0.005274	-0.190321
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	...	-0.009431	0.798278	-0.137458

Which means from this dataset we don’t understand which the transaction attributes are such as user IP location or from how long this is our user. Instead we have numerical values that try to represent that same information. We need to protect the users privacy when we're building systems like this. If you explore the dataset you will be able to see all the columns which include the transaction amount and the class.

Remember each one of the rows represents a user transactions with all the credit and user transaction available. Our objective is to be the classifier that given a new transaction can tell if it's fraudulent or not with a corresponding confidence.

We're not going to build an expert system or rules based system that could for example say if the user IP is from China but that credit card was issued in the U.S. then that transaction must be fraudulent. We're going to build both a deep learning network to try to do so and we're going to apply more traditional machine learning algorithms such as Random forest.

We will build Deep neural network and we will use a stack dense layer. First, we will need to build a classifier which are our classes will determine our network's output. Given our problem we could identify different sort of classes depending on the fraud risk. For example we could cut non fraudulent transactions, risky transactions that did manual remission or just plain simple fraudulent transactions. In our case we'll just work with two possible labels or classifier we have a zero(0) label if the transaction is non fraudulent and on one(1) label if the transaction is fraudulent.

4.2 IMPLEMENTATION

Now we're going to prepare everything we need in our computer so we can do this exercise.
Two of the main tools that we're going to be using are TensorFlow and KERAS TensorFlow will implement everything necessary so we can create and train our deep neural networks and Keras will just make everything easier for us. So in Keras a few lines of code we will be able to create and train our neural network. We will be using both machine learning and deep learning techniques in many occasions those techniques will just complement each other, But in our case, we will just be doing AB testing to see which model performs best. Once we have our deep neural network we're going to just start over using machine learning. For that we're going to be using the Scikit Learn library. It is the most popular machine learning library and it is implemented in Python. We're going to be using Jupyter notebooks to run our python code that code could be given in Amazon Web Services and Google Cloud platform or just your local machine we will be using local machine to load credit card dataset in Jupyter Notebook.

4.3 DATA VISUALIZATION

In Jupyter Notebook we will load the data set (CSV file) after loading necessary libraries such as numpy, pandas, keras and random seed.

In Code below all the necessary libraries and credit card dataset are loaded. You can see the figure 1 to visualize the result of this code.

```
import pandas as pd
import numpy as np
import keras

np.random.seed(2)

data = pd.read_csv('creditcard.csv')

data.head()
```

Figure 4.1: Load libraries and data exploration

We have a total of 28 columns with just numbers that we don't know what they are. This is because our data set has been anonymize. We are not allowed to have real credit card transactions so we can't see that credit card number or where credit card was issued, nor the user IP but that's information are supposed to be in these columns. We also see the amount of the transaction on that last label there are zero if the transaction is non fraudulent and 1 if the transaction is fraudulently.

4.4 DATA PREPROCESSING

Data preprocessing is the first and important step in our case, We have a lot of data preprocessing techniques, In the first step we need to normalize the amount range. In this data set the amount column is increasing from zero to any number.

To normalize this column, we fixed the minimum and maximum range, now it is mapped from -1 to +1, which will be useful for further computation.

In next step we will use the standard scaler function from Scikit learn library to create a new column called are called normalized amount which is important to fit and transform our original amount column and we're going to reshape that to fit to minus one and plus one range. We will also need to drop the original amount column and we are going to use drop function to do that. If we can take a look at our dataset we will realize that we don't have amount column anymore, but

we have normalizeAmount column. We are going to drop time column too which we are not using that column this moment.

Table 4.2: NormalizeAmount column added and drop Time column

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V21	V22	V23	V24	V25	V26	V27	V28	Class	norm
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	0	
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	0	
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	0	
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	0	
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	0	

5 rows × 30 columns

In next step will have to split our data set into the X and Y into the input information that a new transaction will happen which we will try to predict if it's fraudulent or not and the actual label will say if the transaction is fraudulent or not that information is in the last column (class) of the dataset.

We will be using some python functions to do this. We will create an X variable which will get all columns except from column name class from our dataset variable which is called data. And Y variable will contain only class column.

Now X variable have all columns except Class, and Y variable contains only column Class.

4.5 DEEP LEARNING

Machine learning techniques is the science of getting computers work just like humans to extract patterns from information as human do. We do that by feeding them data and information of the world and trying to teach them the underlying patterns and information. Machine can be more efficient and outperform in this task.

Deep learning is the part of machine learning it is based on learning the data presentations that's specific algorithms that are part of the broader machine learning. Deep Learning can be supervised, semi supervised or unsupervised although we could tackle this problem with some unsupervised learning technique, but we will use supervise learning alone, that means that we will need to teach our network with some input and expected output as well.

In our case the input will be a new credit card transaction and the output our label will be the information that says that the input correspond to a fraudulent transaction or not in order to turn

those transactions apart we will do some feature extraction. We're going to try to extract the most relevant information to our classification problem.

Those extracted features will be represented with X1 and X2 as shown in Figure 4.2.

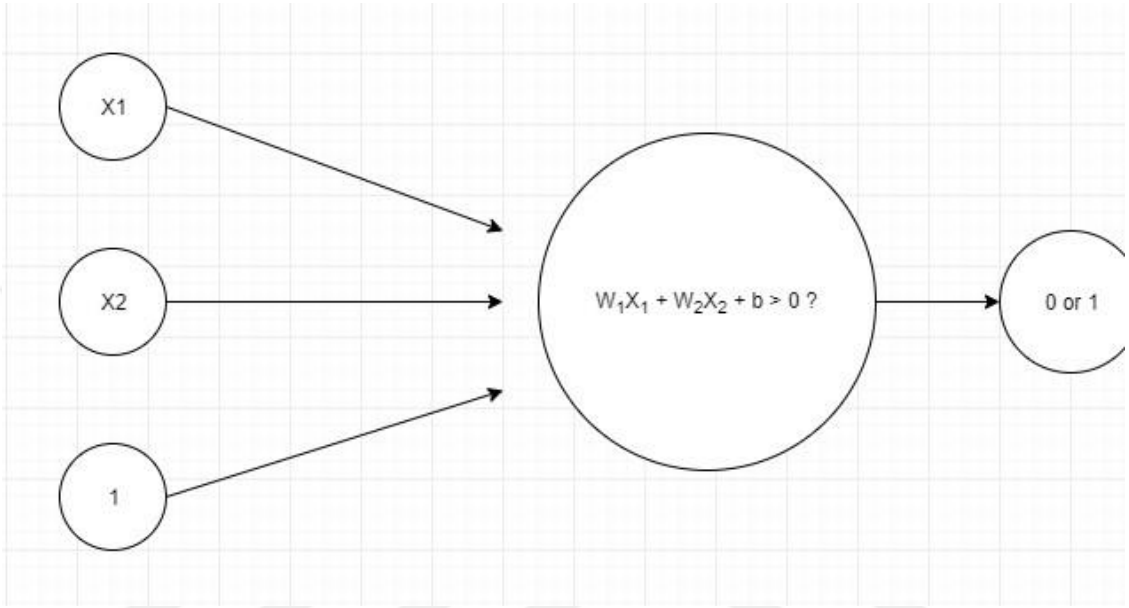

Figure 4.2: Feature Extraction and Perceptron

X1 and X2 can be the users IP location or others that will be the input perceptron that that's the most basic unit in any neural network those numerical inputs will be multiplied by weights represented by W1 and W2, B is just a bias and the output of the equation that you're seeing will be just assume a zero or one for now.

The idea of training our neural network is to find the optimal W1 and W2 and b those optimal values should not only be the best for a single transaction but for the entire data set and ideally twenty new new transaction might come as well.

Just a linear equation will not solve our problem so that’s why we concatenate different perceptron into more complex networks as you can see in Figure 4.3.

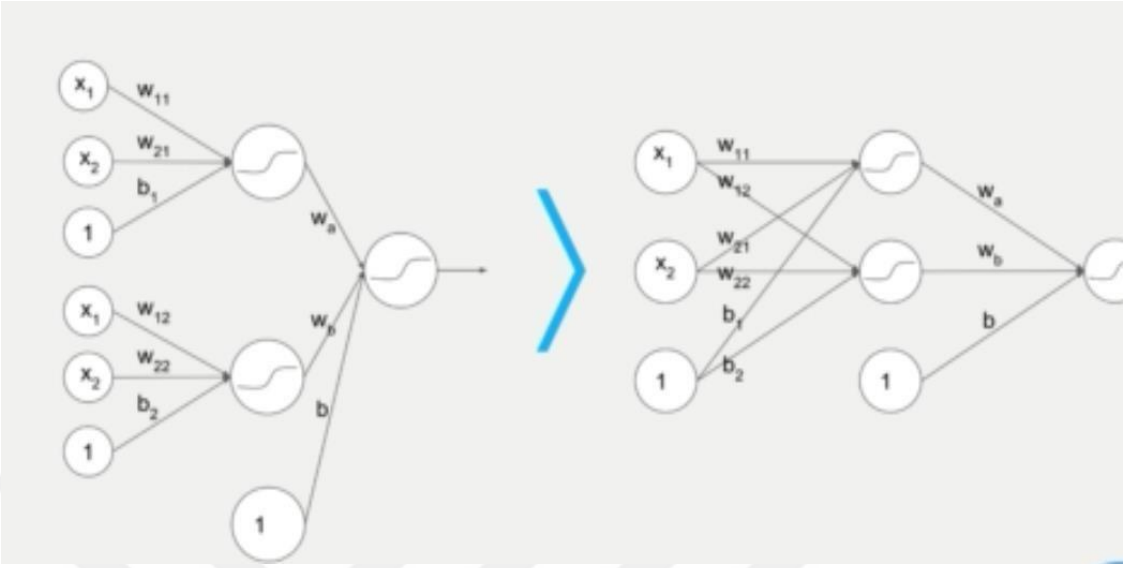

Figure 4.3: Dense Layer

In Figure 4.3 you can see two notations of same architecture, the inputs are always X1 and X2 and weight which we need to train are always W11 and W21 etc. Also notice that how we will put perceptron's together at the same level or set them up at different layers.

Connecting one layers output into another layers input we can create really complex architecture such as the one on Figure 4.4.

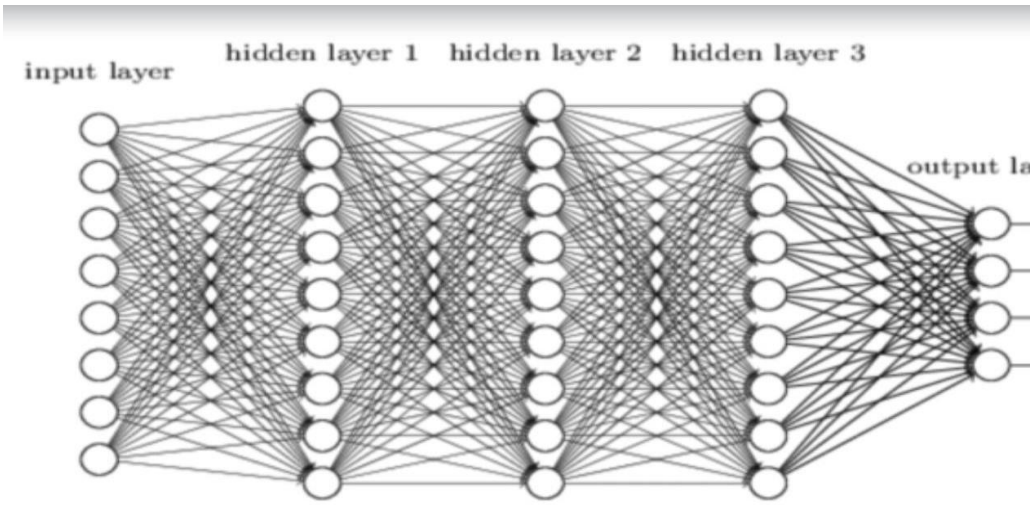

Figure 4.4: Connecting layers

In Figure 4.4 the first layer is called input layer and the last one will be called output layer and all other layers will be called hidden layers.

The output of each perceptron should be 0 or 1 that's why we will be using step function. This function returns 0 for negative values and 1 for positive values. The function that determine the perceptron output are called activation functions and they are important concept in deep learning.

There is some kind of activation function called SIGMOID activation function, It is similar to step activation function but the only difference is that it is softer on the change from 0 to 1

That means that we inject a probability instead of always being certain if our given feature is perceptron or not.

We can now say for example I am 80 percent sure of this output. We will be using it as a parameter in our network.

Another popular activation function is called RELU activation function. RELU stands for

Rectified Linear Unit this function returns a 0 for any negative value and returns same value forany positive values. There are also other activation functions, but we will not use it here. We will stick to STEP, SIGMOID and RELU activation function.

The process of calculating the output for a given input for our neural network is called feedforward. Feedforward is the first step of our training process and it will give us the prediction for our input. We need to compare that with the expected output. Remember that we are using supervised learning techniques. That means that we will have the expected output for any given input that we can use to train our network once we have the expected output and the production together we can see for prediction was right. If we find any difference between them we come back propagate do over through the entire network. That means that we're going to see whose fault was it that we didn't get our predictions right. Once we identified the weights that are causing our mistake we can update them. So our next feedforward process can get a better result than this one we will repeat this process many times until we are satisfied with our results.

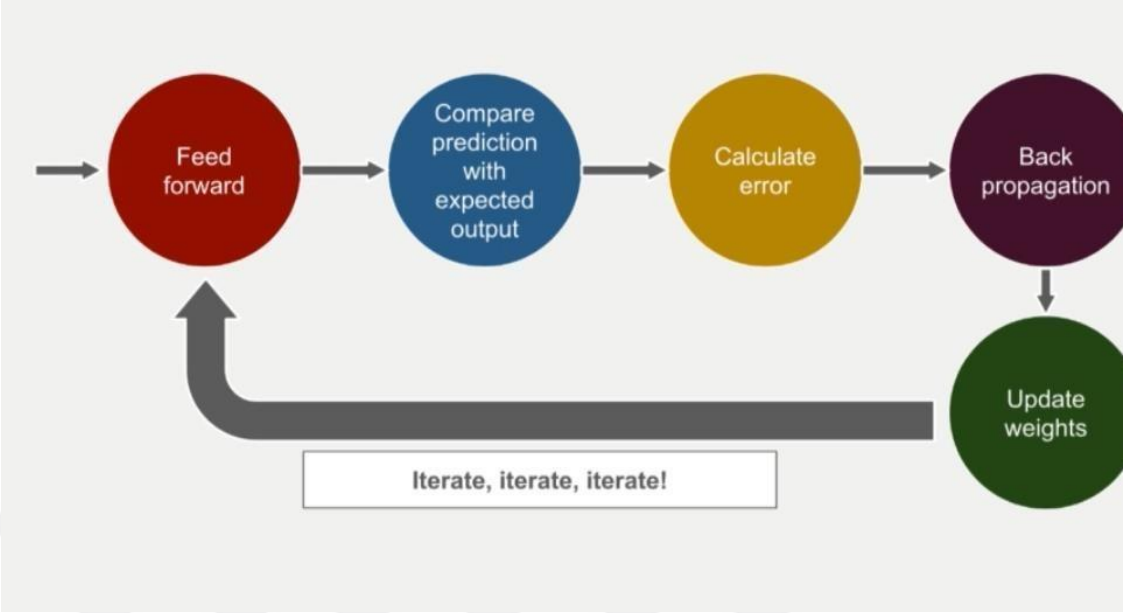

Figure 4.5: FeedForward

But when it will be enough we could determine that we will continue training until we get a given accuracy. But that's hard to do as we don't know how much time will it take us to get there or even if it's possible. So in our case we train for a given amount of a EPOCHS. For example we will iterate over 500 times. This can get to be a really slow process. Suppose we have 1 million records We've got to go over the 1 million records 500 times and that takes a lot of time. The best way is to define a batch size that is the amount of records that we will feedforward at once before doing the right preparation process. That's the slowest part in all the training but we will never train our entire dataset. We will split it into a training, validation and testing dataset that the training dataset will be the one used to feedforward and back propagate through the network. The validation data set will be used to measure objectively results while training. We can use our network's performance on our validation dataset to take decisions about updating it on not and the testing data set is used to evaluate our model after finished training. Testing and validation dataset should be a percentage of training dataset. Which means that if a dataset has 0.1 percent data fraudulent we should cut this fraudulent data in same proportion in three subsets.

The training dataset should be the biggest one which will help us to improve our model. Validation and testing dataset will split from 10 to 40 percent of entire dataset. The goal splitting our dataset is to know how our model will behave with data it never seen during training process. Our model performance must be same for all the datasets, if our model shows underfitting the values we will

add some more layers to our model to reduce underfitting the model. If our model performance show overfitting, then we will use DROPOUT method in which we will remove some random layers to reduce overfitting of the model.

4.6 SPLITTING THE DATA

Here we will implement our model to split the data and the labels into the training data set and the testing data set we will be using the Scikit learn functions to do that.

To import data into train test split function that will produce four outputs which can be label as (x_train, x_test, y_train,y_test). By create X and Y variables for test and train, The size of test can be 30 percent of the data and have fixed random state, Use shape function to understand how much data we have in our test and train data set.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.3, random_state=0)
```

```
X_train.shape
```

```
(199364, 29)
```

```
X_test.shape
```

```
(85443, 29)
```

Figure 4.6: Check the train and test data set

As shown in Figure 4.6 our training dataset have 199364 rows and 29 columns while our testing dataset have 85443 rows and 29 columns after splitting the data.

After splitting the data all the datasets will need to transform in to numpy arrays which is useful and we will need the data in arrays. Figure 4.7 will show the data transformation from dataframes to numpy arrays.

```
X_train = np.array(X_train)
X_test = np.array(X_test)
y_train = np.array(y_train)
y_test = np.array(y_test)
```

Figure 4.7: Data transformation to Arrays

4.7 DEEP NEURAL NETWORK

In this section we will create a deep neural network, that will help us predict if the new transaction is fraudulent or not. To do that we will have to import the sequential model from Keras, Dense layers and dropout layers.

Our model will be a sequential model, that means that one layer will come after the other and we will also define the stack of layers. We will also use a bunch of dense layers and dropout layers in the middle to avoid over fitting as we explain earlier. How many dense layers we must create to a stable result. We will do our first attempt as shown in Figure 4.8 and will measure our results, If we need more complex model we will iterate and create a new model and train the model once more and measure the results again.

We will be using RELU activation function for all dense layers except for the last one, We have SIGMOID activation because we are doing binary classification problem. We will need to tell our deep neural network that how many columns to expect in the first layer that's the input dimension and it must match to the number of columns in our dataset. In our case its 29 so the expected input will be 29.

We will also have a single output node in our last layer of our deep neural network which is the probability of the transaction whether it is fraudulent or not.

We can add as many nodes in all the other layers which we see necessary. We will pass a unit parameter. For the Drop out layer we will add 0.5 probability to dropping each node.

Deep neural network

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
```

```
model = Sequential([
    Dense(units=16, input_dim = 29,activation='relu'),
    Dense(units=24,activation='relu'),
    Dropout(0.5),
    Dense(20,activation='relu'),
    Dense(24,activation='relu'),
    Dense(1,activation='sigmoid'),
])
```

Figure 4.8: Run simple model for first time

Now we have our model defined check the model with model summary function. Check the summary result in Figure 4.9 below.

```
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 16)                480
_________________________________________________________________
dense_2 (Dense)              (None, 24)                408
_________________________________________________________________
dropout_1 (Dropout)          (None, 24)                0
_________________________________________________________________
dense_3 (Dense)              (None, 20)                500
_________________________________________________________________
dense_4 (Dense)              (None, 24)                504
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 25
=================================================================
Total params: 1,917
Trainable params: 1,917
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.9: Model Summary

From Summary function we can see that every parameter we define before are matches. We have almost 2000 parameters to train in our neural network.

4.8 TRAINING

To start training our model we need to call the compile method and use “Adam” optimizer and binary cross entropy loss function and will use accuracy to measure our results and then we will just try the fit method.

We need to pass x_train and y_train variables in to our training method and we need to define a batch size and also define EPOCHS.

In Figure 4.10 we will show the training model results. We will be trying to measure the results of test data in Figure 4.11 and will measure the results and check the performance.

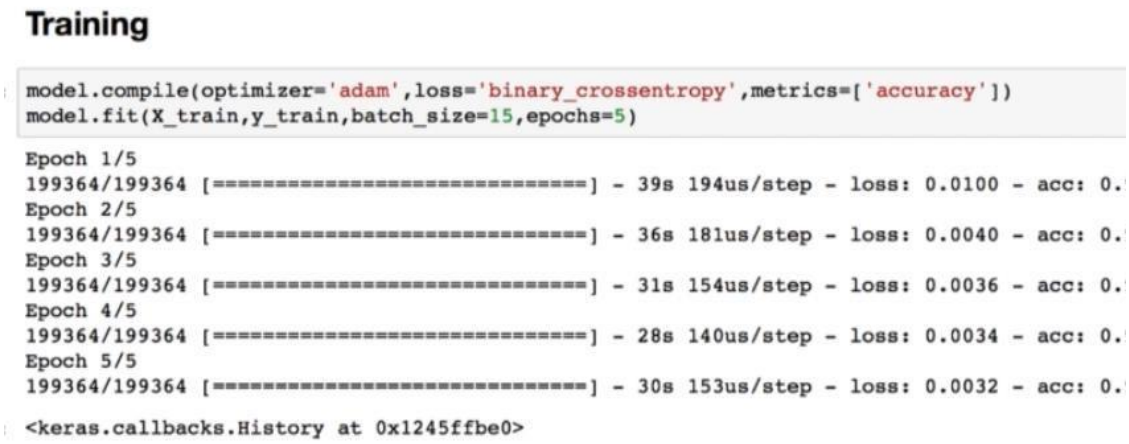

Figure 4.10: Train data results

Now that our network has finish training now we are able to measure the results. As we split our data set in to train and test datasets. Now we will use the x_test and y_test with the evaluate method from Keras. We need to pass the X test and Y test and labels as well so when model predicts the labels we will be able to compare them with the actual expected labels from this input.

```
score = model.evaluate(X_test, y_test)
85443/85443 [==============================] - 4s 46us/step
print(score)
[0.004388740259373431, 0.999403110845827]
```

Figure 4.11: Test Results

We now have the score which means that we 99.94 accuracy in our dataset.

4.9 CONFUSION MATRIX

The confusion matrix show us the performance of our model. In our case we need to know how many fraudulent transactions we did that correctly if we only measure the percentage of their correct labels we will get a really high value as most of our transactions are known for non-fraudulent. If our model always predicts non fraudulent we will just get the majority of them righ
t.

Table 4.3 Confusion Matrix

		True Labels	
Prediction		**Non-Fraudulent**	**Fraudulent**
	Non-Fraudulent	True Negative	False Negative
	Fraudulent	False Positive	True Positive

We are going to see how many true positives we have which means how many fraudulent transactions we predicted as fraudulent. Also with confusion matrix we will measure how many

true negatives we have which means how many non-fraudulent transactions we predicted as non-fraudulent. How many false positive which means how many fraudulent transaction as non-fraudulent these are the users which will be able to steal from us, those are the ones which we will not be able to detect and that's the number which want to be as low as possible. For last, how many false negative we have that's how many we predicted as fraudulent although they were good users, that's for sure be unsatisfied users as they were willing to pay but we didn't allow them. In our given problem we will try to get as many true negative and true positive as possible and also try to reduce as much as possible the number of false negative and false positive. In our application we will prefer to increase the number of FP as long as we decrease the number of FN. This might be different in other applications.

Besides the confusion metrices we can use other metrics to try to measure the behavior of our model. The most frequent metrices is accuracy(4.1), precision(4.2),specificity(4.3) and recall(4.4).

$$\textbf{Accuracy Metric} = \frac{TP + TN}{TOTAL} \tag{4.1}$$

$$\textbf{Precision Metric} = \frac{TP}{TP + FP} \tag{4.2}$$

$$\textbf{Specificity Metric} = \frac{TN}{FP + TN} \tag{4.3}$$

$$\textbf{Recall Metric} = \frac{TP}{TP + FN} \tag{4.4}$$

That would mean that the amount of transaction that we could correctly detect as fraudulent and we will try to increase this number as much as possible.

When we start training we will try to increase our Recall as much as our Accuracy that would mean that our model is performing well.

To try the confusion matrix in the code and to check our model how our confusion matrix is behaving, we will use standard code for this from Scikit Learn library which is already available and is shown in Figure 4.12. To plot the confusion matrix from scikit learn library we will need to import Matplotlib.pyplib library too.

```
import matplotlib.pyplot as plt
import itertools

from sklearn import svm, datasets
from sklearn.metrics import confusion_matrix

def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

Figure 4.12: Confusion matrix code

For plotting our confusion matrix for our model, first we will be going to need the prediction for our x_test input and we will use the predict method and will applied to our model. We will also need to transform our y_test variable to a panda dataframe that's the needed input for our confusion matrix. We will be able to create our confusion matrix. We will need to pass the y_test expected output and the predictions y_pred values to the confusion matrix function. Now we can print the confusion matrix. In figure 4.13 we will create the confusion matrix and you can see the results in figure4.14.

```
y_pred = model.predict(X_test)
y_test = pd.DataFrame(y_test)
```

```
cnf_matrix = confusion_matrix(y_test, y_pred.round())
```

```
print(cnf_matrix)
```

```
[[85281    15]
 [   36   111]]
```

```
plot_confusion_matrix(cnf_matrix, classes=[0,1])
```

```
Confusion matrix, without normalization
[[85281    15]
 [   36   111]]
```

```
plt.show()
```

Figure4.13: Create confusion matrix

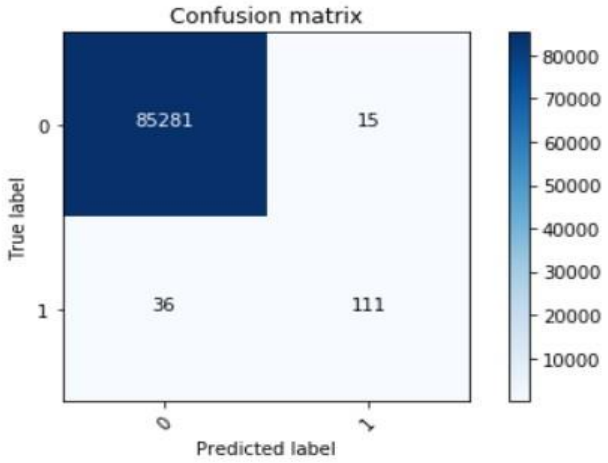

Figure 4.14: Confusion matrix results

As we can see in Figure 4.14 that almost every non-fraudulent transactions we detect correctly and we got a lot of fraudulent transactions but we still have a lot of fraudulent transactions that we mark as non-fraudulent which is 36 which means 36 out of 111 users were able to steal from us .

Lets plot the same confusion matrix but for entire data set later we will apply some different sampling techniques and will get some objective measures that how our model is performing.

See figure 4.15 confusion matrix for entire dataset

```
y_pred = model.predict(X)
y_expected = pd.DataFrame(y)
cnf_matrix = confusion_matrix(y_expected, y_pred.round())
plot_confusion_matrix(cnf_matrix,classes=[0,1])
plt.show()
```

```
Confusion matrix, without normalization
[[284248     67]
 [   105    387]]
```

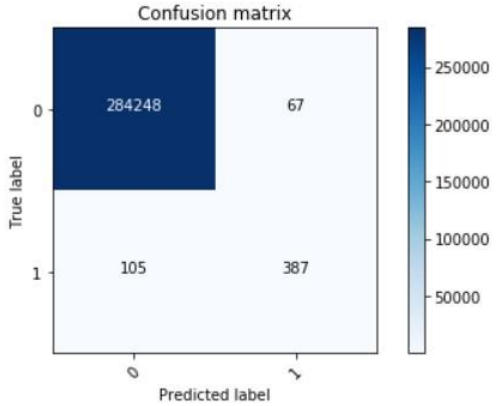

Figure 4.15: Entire data set confusion matrix results

We come over with more than 100 fraudulent transactions that got an incorrect label.

4.10 MACHINE LEARNING CLASSIFIERS

Here we will try to solve our classification problem with different machine learning algorithms. We are using classification because we are not dealing with continuous bias. Instead we are dealing with discreet bias which is fraudulent and non-fraudulent transactions. The most popular algorithms is SVM Support vector machines. SVM will try to answer which is the best line to split the two classes. The best line would be the one which maximize the probability of getting a new point and label it correctly.

The other algorithm we will use is Decision trees which is very commonly use in machine learning to solve classification problems. If we apply decision trees to our specific problem the output would be the fraudulent transactions or non-fraudulent transactions, we would be analyzing the transactions features such as user IP location or home address etc. Decision trees can get to be very efficient, they will stop analyzing once they found the terminal node, the key to build a good decision tree is to find a good feature to analyze in each decision node. Decision trees can solve classification and regression problems which means they can work with discreet and continuous bodies. Trees also build recursively and they use cost functions to do that. Cost functions is trying

to find the best path to split the data at each node, so they can get faster to the solution and not waste time asking questions that won't get us any closer to reaching termination node. There is also a technique called Pruning that is used to avoid the overfit.

Another technique we could apply is called random forest. This is the usage of more than one decision trees combine at once. The same input is run over different decision trees that are created randomly as we already know the expected output, We can see which decision trees get to the best solution from the expected answer from all their randomly created once. If we have a new input that's not in our training data we can go over the same decision trees we found very important to find prediction. This can be really naive but it has proven to have really good experimental results.

Another popular algorithm is called KNN K Nearest Neighbors. This algorithm is simple and can solve the problems with multiple dimensions. We have seen so far some of the most frequent classification algorithms and following this we will apply just a couple of them to try to solve our problem.

4.11 RANDOM FOREST

Using Random forest algorithm, we are trying to import our result which we will show in the confusion matrix. We will need to import this algorithm RandomForestClassifier from sciket learn library which is already programmed. We will need to indicate on parameter that’s the number of estimators in a way how complex our random forest would be. We will also call that fit method which is for training where (x_train) and (y_train) variables will be used and will need to transform y_train into a valid input. We also define y_pred variable and use random forest to train to predict on test variable.

We will also calculate the score for the accuracy we will need the x_test and y_test variable for that. Now will plot the confusion matrix to see the results.

Figure 4.16 have random forest algorithm code and figure 4.17 will show the confusion matrix.

Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

```
random_forest = RandomForestClassifier(n_estimators=100)
```

```
random_forest.fit(X_train,y_train.values.ravel())
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

```
y_pred = random_forest.predict(X_test)
```

```
random_forest.score(X_test,y_test)
```

```
0.9995435553526912
```

Figure 4.16: Random Forest Algorithm

```
cnf_matrix = confusion_matrix(y_test,y_pred)
```

```
plot_confusion_matrix(cnf_matrix,classes=[0,1])
```

```
Confusion matrix, without normalization
[[85290     6]
 [   33   114]]
```

```
plt.show()
```

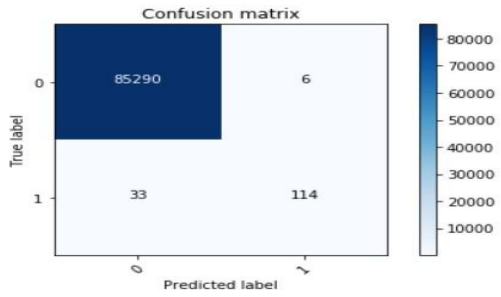

Figure 4.17: Random Forest results in confusion matrix

As we see from random forest confusion matrix result that we have a really low amount of confuse items but still we can improve. Let's try this algorithm with entire dataset and check the results in figure 4.18.

```
y_pred = random_forest.predict(X)

cnf_matrix = confusion_matrix(y,y_pred.round())

plot_confusion_matrix(cnf_matrix,classes=[0,1])

Confusion matrix, without normalization
[[284309      6]
 [    34    458]]

plt.show()
```

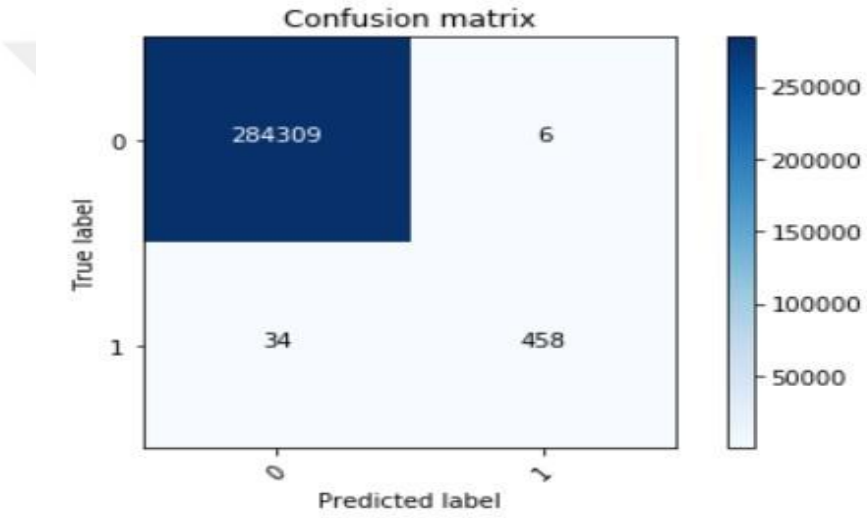

Figure 4.18: Random Forest Entire data set Results

Now we have only 34 mis label transactions from entire data set.

4.12 DECISION TREES

Now we're going to go over the same problem. We're going to be using the same first part of the Jupiter notebook we used in tests we're going to apply decision trees. That's another possible machine learning classification algorithm. We will first import that decision tree classifier that is already implemented for us in the Scikit Learn library. We will also create a new decision tree classifier and put that into a variable called Decision Tree. We will also trying using fit method same as before for this we will need the x train y train variables. We will need to transform the Y train variable into a valid input. We will define y_pred variable using the predict method in the decision tree. We will apply this to the x test variable. We can do this to measure our performance in our test dataset. We will use the score method that will need the X test and y test variables but we will also want to see our confusion matrix. We just use the same code as before to get the prediction for the entire data set using the same predict method. We will get the expected y in pandas dataframe then we will create the confusion matrix with this information we also need to round our predictions just always and after this we can call our plot confusion matrix function. As you can see in figure 4.19.

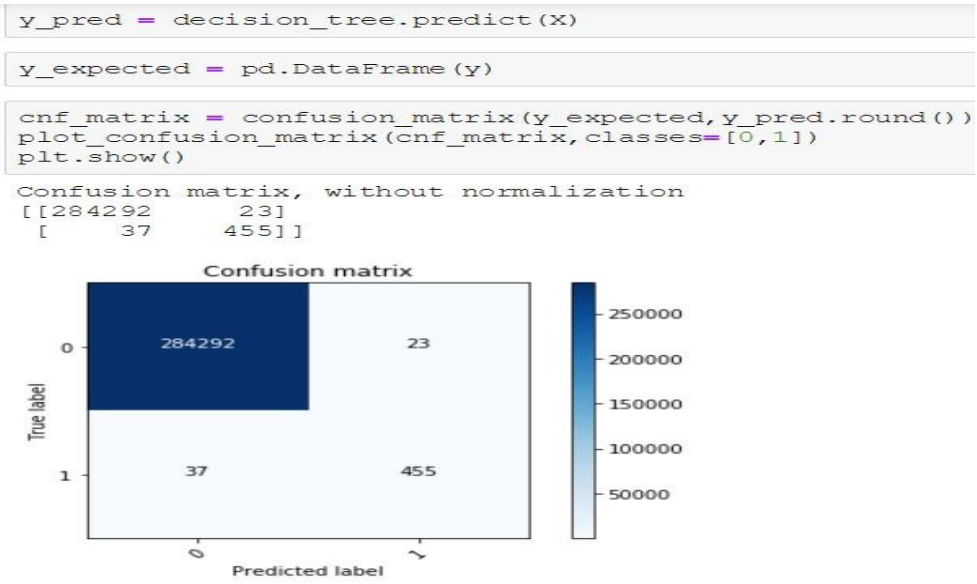

Figure 4.19: Decision Tree Results

It seems that we got really good results only 47 out of the almost 500 transactions were incorrectly label.

4.13 UNDER SAMPLING TECHNIQUES

We going to apply some under sampling techniques. We'll start by locating our fraudulent transactions indexes in our dataset will also be interested in counting how many fraudulent indexes we have. Now we will do the same for the non fraudulent transactions. Everything will be the same except for the class that we will be searching. Now from the non-fraudulent indexes will just select random sample matching the same amount of fraudulent transactions that we have. We also need to transform into a numpy array now to verify we'll check the length of the new non-fraudulent transactions selected. We just create a new array with all the indexes of fraudulent and non-fraudulent transactions just by concatenating both indexes to verify we will check the length of the entire new array, now we have the selected indexes for under sampling we just select the corresponding rows will do the same as before just to split our data into x and y variables. We'll also split our data into train on test data sets. Now will transform everything to numpy arrays. with just verify that our model remains unchanged. We can do that with summary function same as before.

We will plot our confusion matrix on entire dataset to check the performance of the model after under sampling. Check figure 4.20 to see the result in confusion matrix after under sampling.

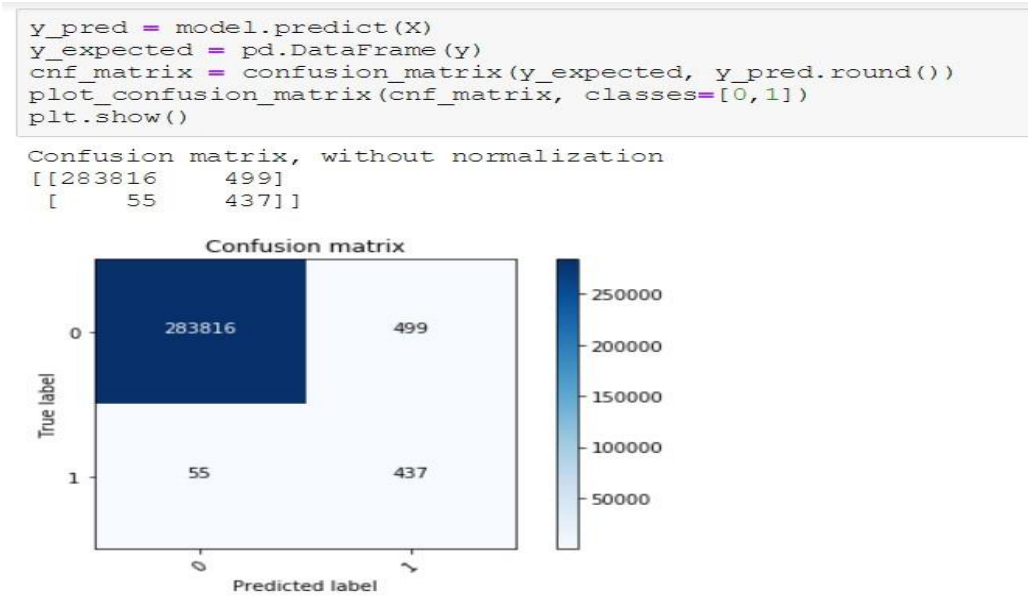

Figure 4.20: Under sampling technique

4.14 SMOTE

We get the idea that sometime losing data to train is not the best way to go. What we're going to do is to apply some over sampling techniques such as SMOTE. First, we need to install smote before going forward. To use SMOTE X and Y variables will be needed to give it as input to fit function. We will need to transform the output of this function in to pandas data frames which we will use it later. Same split function will use to split our dataset in to test and train datasets. We will retrain our model with the new data.

Now we will plot our confusion matrix for our train dataset and then for the complete dataset. Figure 4.21 will show the confusion matrix of the entire data set.

```
y_pred = model.predict(X)
y_expected = pd.DataFrame(y)
cnf_matrix = confusion_matrix(y_expected, y_pred.round())
plot_confusion_matrix(cnf_matrix, classes=[0,1])
plt.show()
```

```
Confusion matrix, without normalization
[[283249   1066]
 [     0    492]]
```

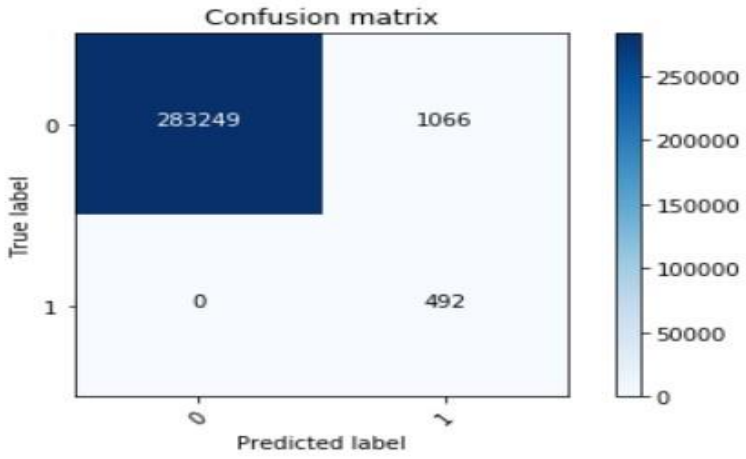

Figure 4.21: Smote confusion matrix

Notice that with this technique we have no non-fraudulent transactions which label as fraudulent. Which means that we don’t have false positive which reduces the amount of work for fraud department in ecommerce business.

5. CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

In this section we will conclude the whole work we done on data preprocessing and on real life ecommerce business problem. We proposed several models and check the behavior of the model and then compare the results of different datasets to check the importance of data preprocessing. Without data preprocessing or not focusing that much on data preprocessing can have huge effect on the results. As we seen in our last model we have the high accuracy after we remove the missing values and remove the outliers from the datasets which are the main important parts of machine learning.

In this thesis we start focusing on importance of data preprocessing in machine learning which is nowadays trending in the scientific research work and companies all over the world invest a lot of money on data. There are some phrases like "The data will be the next oil in the world". Most important issues in the recent era is the credit card fraud which we tried to prove it with machine learning techniques focusing on data preprocessing which the first step of data mining or machine learning. We studied about missing data in the large data sets, It is very common that most of the large datasets have some missing values which sometimes effect the results of the models, as we explain in chapter two how to take action on missing data which is called data cleaning, we proposed two solutions one is to just remove the missing data if the data is too big in that case some data removal will not effect the results and other solution is to take the average of the data and fill the values. We also find that we have some outliers in our data which also very important to deal with outliers, the techniques like data scaling and data centering is the most routine solution in data transformation which helps to remove outliers from the data.

We use python programming language for all our computation in this thesis. Some important libraries we imported to make our work easy and authentic.

We tried to solve our classification problem with different machine learning algorithms. Like decision trees, random forest and Under sampling. We train and test our models with these different techniques and in the end, we compare the results.

In random forest, we run this algorithm on our train and test data sets and we find that in the result of this algorithm we found the confuse values are become decrease to 33.

We run the same tests on the decision tree algorithm and we found the confuse values are 37 in this case the random forest algorithm is more efficient on our dataset.

In last algorithm we preprocess our data which was the main objective of this thesis, by preprocessing the data I mean we split the fraudulent transactions equally in both our data sets train and test dataset in same ratio and this time we use some over sampling techniques like SMOTE and our results were very impressive. In our last test we found there was no confuse values in confusion matrix. Which means that we reduce the big amount of work for the fraud department in any ecommerce business.

All our experiments results show that our initial hypothesis about data preprocessing role is true and as we see in our last experiment after data preprocessing for the model give us high accuracy from all other experiments. This thesis explains all the structure of the process how to build the model and prepare data and how to build a model to perform data analysis which I believe it can be easily put into practice.

5.2 FUTURE WORK

Future developments can be possible in data preprocessing techniques and create a model with more features, the results will be more effective and real. We also need to work on the computational time of preparing the data and passing data through the model. Normally in our case our credit card fraud data was taking to much time to give us the results, In future work we will need to reduce the time.

It will be more interesting that we apply this model and data preparation technique in market on larger dataset to see the performance. For example, we can apply this to any ecommerce business online who accept the credit card transactions to have a check on the fraudulent transactions.

REFERENCES

[1] Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Comput. System Sci 76(1)(2010), 3-20.

[2] Breiman L., Friedman J. H., Olshen R., Stone C. J.: Classification and Regression Trees. Chapman & Hall, NY, 1984.

[3] Fu H., Fu H., Njiwoua P., Mephu Nguifo E.: A comparative study of FCA-based supervised classification algorithms. In: Proc. ICFCA 2004, LNAI 2961, 2004, pp. 313–320.

[4] K. K. Srinivas, G. R. Rao, and A. Govardhan. Analysis of coronary heart disease and prediction of heart attack in coal mining regions using data mining techniques. In The 5th International Conference on Computer Science & Education, pages 1344–1349, 2010.

[5] K. Rothaus, X. Jiang, T. Waldeyer, L Faritz, M. Vogel, and P. Kirchhof. Data mining for detecting disturbances in hearth rhythm. In Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, Kunming, pages 3211–3216, 2008.

[6] Y. Xing, J. Wang, and Z. Zhao. Combination data mining methods with new medical data to predicting outcome of coronary heart disease. In International Conference on Convergence Information Technology, pages 868–872, 2007.

[7] Dash, M. and Liu, H., "Feature selection for classification," Intelligent Data Analysis, Vol. 1, No. 3, 1997, pp. 131-156.

[8] Fu, Y. J., "Data mining," IEEE Potentials, V.16, Issue 4, Oct.-Nov. 1997, pp. 18-20.

[9] Han, J. W., "Data Mining: Where Is It Heading?" Proc. of 13th Mt. Conf. on Data Engineering, 1997, pp. 508-508.

[10] Kelly, J. D. and Davis, L., "Hybridizing the genetic algorithm and the K-nearest neighbors classification algorithm", Proc. of the fourth Int. Conf. on Genetic Algorithms and their Applications, 1991, pp. 377-383.

[11] Lin, J., Hwang, M. & Becker, J. “A Fuzzy Neural Network for Assessing the Risk of Fraudulent Financial Reporting” Managerial Auditing Journal 18(8) (2003): 657-665

[12] Bell, T. & Carcello, J. “A Decision Aid for Assessing the Likelihood of Fraudulent Financial Reporting” Auditing: A Journal of Practice and Theory 10(1) (2000): 271- 309.

[13] Fanning, K., Cogger, K. & Srivastava, R. “Detection of Management Fraud: A Neural Network Approach”. Journal of Intelligent Systems in Accounting, Finance and Management 4 (1995): 113-126.

[14] Summers, S. & Sweeney, J. “Fraudulently Misstated Financial Statements and Insider Trading: An Empirical Analysis” The Accounting Review January (1998): 131-146.

[15] Beneish, M. “Detecting GAAP Violation: Implications for Assessing Earnings Management Among Firms with Extreme Financial Performance” Journal of Accounting and Public Policy 16 (1997): 271-309.

[16] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In Proceedings of the genetic and evolutionary computation conference, pages 11–18, 2002.

[17] Andrea Dal Pozzolo, Olivier Caelen, Serge Waterschoot, and Gianluca Bontempi. Racing for unbalanced methods selection. In Proceedings of the 14th International Conference on Intelligent Data Engineering and Automated Learning. IDEAL, 2013.

[18] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson, and Gianluca Bontempi.

Calibrating probability with undersampling for unbalanced classification. In 2015 IEEE Symposium on Computational Intelligence and Data Mining. IEEE, 2015.

[19] Tom M Mitchell. Machine learning. 1997. Burr Ridge, IL: McGraw Hill, 45, 1997.

[20] Michalski R. S.: A theory and methodology of inductive learning. Artificial Intelligence20(1983), 111–116.

[21] Missaoui R., Kwuida L.: What Can Formal Concept Analysis Do for Data Warehouses? In Proc. ICFCA 2009, LNAI 5548, 2009, 58–65

.

[22] Murphy P. M., Pazzani M. J.: ID2-of-3: constructive induction of M-of-N concepts for discriminators in decision trees. In Proc. of the Eight Int. Workshop on Machine Learning, 1991, 183–187.

[23] Murthy S. K., Kasif S., Salzberg S.: A system for induction of oblique decision trees. J. of Artificial Intelligence Research 2(1994), 1–33.

[24] Pagallo G., Haussler D.: Boolean feature discovery in empirical learning. Machine Learning 5(1)(1990), 71–100.

[25] Piramuthu S., Sikora R. T.: Iterative feature construction for improving inductive learning algorithms. Expert Systems with Applications 36(2, part 2)(2009), 3401–3406.

[26] S. Kullback and R. A. Leibler. On information and sufficiency. The Annals of Mathematical Statistics, 22(1):79–86, 1951.

[27] http://weka.sourceforge.net/doc.dev/weka/attributeselection/ranker.html.

[28] Mark A. Hall. Correlation-based Feature Selection for Machine Learning. PhD thesis, University of Waikato, 1999.

[29] http://weka.sourceforge.net/doc.dev/weka/attributeselection/greedystepwise.html.

[30] Ian H. Witten and Eibe Frank. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[31] Kohonen, T, "Self-organized formation of topologically correct feature maps," Biological Cybernetics, Vol. 43, 1982, pp. 59-69.

[32] Kohonen, T, "The self-organized maps," Proc. of the IEEE, Vol. 78, No. 9, 1990, pp. 1464-1480. 67 68

[33] Langley, P. and Blum, A. L., "Selection of relevant features and examples in machine learning," Artificial Intelligence, Vol. 97, No. 1-2, Dec. 1997, pp. 245-271.

[34] Lu, H. J., Setiono, R. and Liu, H., "Effective data mining using neural networks," IEEE Trans. on Knowledge and Data Engineering, Vol. 8, Issue 6, Dec. 1996, pp. 957-961.

[35] Gallant. "Nonlinear regression." The American Statistician 29.2 (1975): 73-81.

[36] Bishop, Christopher M. Neural networks for pattern recognition. Oxford university press, 1995.

[37] Linggard, Robert, D. J. Myers, and C. Nightingale, eds. Neural networks for vision, speech and natural language. Vol. 1. Springer Science & Business Media, 2012.

[38] Kruse, Rudolf, et al. "Multi-Layer Perceptrons." Computational Intelligence. Springer London, 2013. 47-81.

[39] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." Cognitive modeling 5 (1988).

[40] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. No. ICS-8506. CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR COGNITIVE SCIENCE, 1985.

[41] Smolensky, Paul. "Information processing in dynamical systems: Foundations of harmony theory." (1986): 194.

[42] Cortes, C., Pregibon, D. & Volinsky, C. “Computational Methods for Dynamic Graphs” Journal of Computational and Graphical Statistics 12 (2003): 950-970.

[43] Cahill, M., Chen, F., Lambert, D., Pinheiro, J. & Sun, D. “Detecting Fraud in the Real World” Handbook of Massive Datasets (2002): 911-930.

[44] Rosset, S., Murad, U., Neumann, E., Idan, Y. & Pinkas, G. “Discovery of Fraud Rules for Telecommunications - Challenges and Solutions” Proc. of SIGKDD99, (1999): 409- 413.

[45] Kim, H., Pang, S., Je, H., Kim, D. & Bang, S. “Constructing Support Vector Machine Ensemble” Pattern Recognition 36 (2003): 2757-2767.

[46] Burge, P. & Shawe-Taylor, J. “An Unsupervised Neural Network Approach to Profiling the Behavior of Mobile Phone Users for Use in Fraud Detection” Journal of Parallel and Distributed Computing 61 (2001): 915-925.

[47] Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams.

In Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 97–106. ACM, 2001.

[48] João Gama, Ricardo Fernandes, and Ricardo Rocha. Decision trees for mining data streams. Intelligent Data Analysis, 10(1):23–45, 2006.

[49] Jing Gao, Wei Fan, Jiawei Han, and S Yu Philip. A general framework for mining concept-drifting data streams with skewed distributions. In SDM, 2007.

[50] Jing Gao, Bolin Ding, Wei Fan, Jiawei Han, and Philip S Yu. Classifying data streams with skewed class distributions and concept drifts. Internet Computing, 12(6):37–49, 2008.

[51] Gregory Ditzler and Robi Polikar. An ensemble based incremental learning framework for concept drift and class imbalance. In Neural Networks (IJCNN), The 2010 International Joint Conference on, pages 1–8. IEEE, 2010

[52] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[53] O. Bayat, B. Shafai, and O. N. Ucan, "An Efficient Channel Equalization on the Transmission of Turbo Coded Signals," p. 7.

[54] T. A. Mohammed, S. Alhayali, O. Bayat, and O. N. Uçan, "Feature Reduction Based on Hybrid Efficient Weighted Gene Genetic Algorithms with Artificial Neural Network for Machine Learning Problems in the Big Data," *Scientific Programming*, 2018. [Online]. Available: https://www.hindawi.com/journals/sp/2018/2691759/abs/. [Accessed: 07-Jul-2019].

[55] H. Al-Rayes, "Feature Selection using Salp Swarm Algorithm for Real Biomedical Datasets," Mar. 2018.

[56] S. Al-hayali, O. Ucan, and O. Bayat, "Genetic Algorithm for Finding Shortest Paths Problem," in *Proceedings of the Fourth International Conference on Engineering & MIS 2018*, New York, NY, USA, 2018, pp. 27:1–27:6.

[57] T. A. Mohammed, O. Bayat, O. N. Uçan, and S. Alhayali, "Hybrid Efficient Genetic Algorithm for Big Data Feature Selection Problems," *Found. Sci.*, Mar. 2019.

[58] T. A. Mohammed, A. alazzawi, O. N. Uçan, and O. Bayat, “Neural Network Behavior Analysis Based on Transfer Functions MLP & RB in Face Recognition,” in *Proceedings of the First International Conference on Data Science, E-learning and Information Systems*, New York, NY, USA, 2018, pp. 15:1–15:6.

[59] M. I. A. Al-Mashhadani, O. N. Ucan, and O. Bayat, “Melanoma Cancer Analysis Towards Early Detection Using Machine Learning Algorithms,” Mar-2019. [Online]. Available:https://www.ingentaconnect.com/contentone/asp/jmihi/2019/00000009/00000003/art00017. [Accessed: 07-Jul-2019].

[60] A. K. Abbas, T. A. Mohammed, O. Bayat, and O. N. Ucan, “The prediction of fusion degree of international groups from their Twitter accounts,” in *2017 International Conference on Engineering and Technology (ICET)*, 2017, pp. 1–8.

[61] S. Q. F. Albawi, “SOCIAL TOUCH GESTURE RECOGNITION USING DEEP NEURAL NETWORK,” p. 130.

[62] S. Q. Fleh, O. Bayat, S. Al-Azawi, and O. N. Ucan, “A Systematic Mapping Study on Touch Classification,” p. 10, 2018.