Anomaly Detection in Time Series Dataset

Atul Kumar Jha

A Dissertation

Presented to the University of Dublin, Trinity College

in partial fulfilment of the requirements for the degree of

Master of Science in Computer Science (Data Science)

Supervisor: John Waldron

September 2020

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Atul Kumar Jha

September 7, 2020

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Atul Kumar Jha

September 7, 2020

Acknowledgments

First of all, I would like to express my sincere gratitude to my advisor, Dr. John Waldron, for his consistent support and guidance during the planning and development of this research work. I am also thankful to Computer Science and Statistics Department of Trinity College Dublin for providing me with infrastructure and the finest environment to work on my thesis.

I would also like to thank my second reader Dr. Bohman Honari for his valuable time and feedbacks which helped me acutely.

I would also like to thank all my friends and classmates for all the great memories that we shared together and for making this entire college journey awesome here at Trinity College Dublin.

Atul Kumar Jha

University of Dublin, Trinity College
September 2020

Anomaly Detection in Time Series Dataset

Atul Kumar Jha, Master of Science in Computer Science

University of Dublin, Trinity College, 2020

Supervisor: John Waldron

In terms of data mining, anomalies are defined as a data object that vary significantly from the rest of the values, as if it was generated by a different mechanism. Early detection of anomalies plays a vital role in any organisation. To maintain the consistency of an individual's data and to protect any corporation against malicious attacks, it is important to detect anomalies as early as possible. Due to security and privacy issues there are very limited real-world dataset available that can be used to benchmark the anomaly detection models based on their test score. The available datasets are the both real and synthetic some of them being highly imbalanced. Here the paper purposes different kind of approach which can be used when dealing with the imbalanced dataset while maintaining the integrity of the dataset as well as to differentiate which kind of algorithms works best on different types of dataset.

In this study, various methods have been performed that can be used when working on anomaly detection, as well as comparing their efficiency. The proposed system uses different algorithms like Linear Regression, Ridge Regression, ARIMA model, Long-Short-Term-Memory, Gaussian Distribution method, Auto regression model, etc. For evaluation purpose different metrics such as confusion matrix, ROC curve, precision, recall were used. The result showed that K-means model score was able to maintain good score on both Yahoo S5 and Numenta dataset, However ARIMA worked well only on Yahoo S5 dataset. For the Credit Card dataset, the most difficult part was to balance the dataset once balanced, the result showed that Logistic regression worked well on it whereas other algorithms struggled to classify correctly.

Summary

The main purpose of this dissertation is to use machine learning to detect and predict any kind of anomalies in time-series dataset while taking care of the nature of highly imbalanced dataset which may result into overfitting or may result into high number of False Negative values. The project is divided into two parts, the 1st is the process of cleaning and modifying the dataset in such a way that the highly imbalanced dataset can be transformed into a balanced dataset while maintaining the integrity of the dataset. The 2nd part is to use that dataset and to apply different algorithms to both detect and predict any kind of anomalies in time-series dataset and to create a benchmark of the models in respect to the different datasets.

To deal with the highly imbalanced dataset we used the Random Under-Sampling method which basically means that we have to remove some part of the dataset to make the overall dataset more balanced which will help us to avoid model overfitting. The few dataset which were used in this project are Yahoo S5 - A Labeled Anomaly Detection Dataset, version 1.0, Credit Card Transactions by European cardholders and The Numenta Anomaly Benchmark (NAB). For the detection of anomalies different machine learning algorithms have been used such as ARIMA, K-means, Logistic Regression etc. For analysis purpose and to simulate the real-time flow of data we have used Tableau. The benchmark have been set using different metrics such as Area under ROC curve, f1 score, confusion matrix etc.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Anomalies are identified as anything that deviates from the normal behaviour or does not conform to the common pattern. Anomaly detection is important because the anomalous items translate into significant and alarming information in a broad variety of application domains. E.g., anomalies in bank transactions data could signify that credit card fraud has taken place. Anomalies in medical diagnostic images may indicate the presence of a disease or malignant tumours. Anomalies in BAN (Body-Area-Network) sensor data can be analyzed to identify early health warnings. Currently the main application of anomaly detection are being used in the banking sector because of dramatically rise of electronic bank transactions such as credit card uses in every shopping merchandise which results into a huge spike in fraudulent transactions. Several Fraud Detection Systems are now using machine learning algorithms to detect any kind of anomalies to to prevent any fraudulent activity from happening. But according to the data provided by the European Bank, in every 100,000 there are only 1000 fraud transactions (anomaly labels) which makes the data highly imbalanced which may affect the model accuracy when it comes to detecting or predicting any anomalies.

1.2 Challenges

An anomaly is a complete nightmare for any organization, several companies are struggling with eliminating the anomalies from the datasets. Due to the nature of dataset which is highly imbalanced the machine learning models struggles a lot to predict the anomalies correctly which results into large number of false negative or false positive values. The datasets are highly imbalanced because the number of labelled anomalies are quite less when compared to the overall dataset. Moreover, in our case if the dataset is real-time streaming dataset then the model needs to be prepared to both process the dataset as well as detect any presence of anomaly in real-time.
Another main challenge in anomaly detection is to find a robust classification and detection method that avoids false positive to minimum and maintain high accuracy.

1.3 Purpose

Our aim is to work on the streaming or time-series dataset and to pre-process the dataset to transform it to a balanced dataset and to come up with different algorithms and strategies to detect anomalies using machine learning without any human efforts. A single model is not enough for the anomaly detection, different kind of dataset requires different algorithms for anomalies to be detected. So we used three different kind of dataset to compare the models and to create an accurate benchmark based of several evaluation metrics.

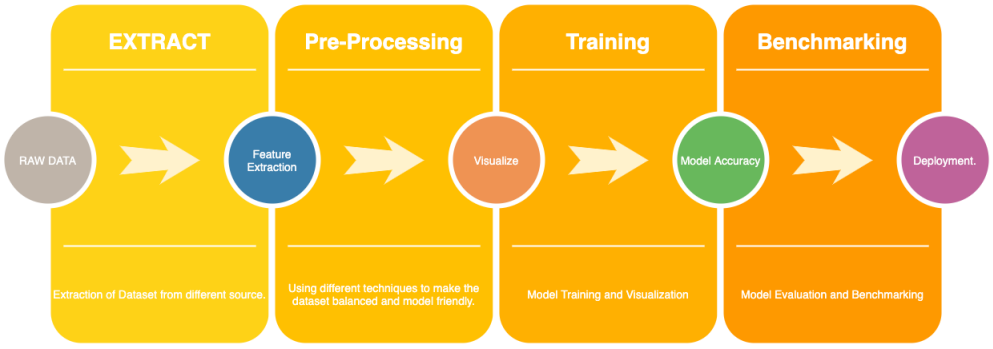


Figure 1.1: Working Model Pipeline

1.4 Motivation

Anomalies are like a virus which underlie in between any particular process and can seriously damage the end results. For e.g, an anomaly in bank transactions can signify any unusual transactions, or an anomaly in a routine medical x-ray can give an indication of a bigger health problems. In both cases if these anomalies can be detected earlier, can stop the damage to be done before its too late. But detecting anomalies are not that easy because of the nature of the dataset. Anomalies doesn't appear often which makes any recorded dataset being highly imbalanced and biased. Our aim here is to provide an optimal solution to detect any kind of anomaly using machine learning while minimising any possible False Negative values, which simply means being as accurate as possible when it comes to detection of anomalies without any human effort. Several studies have already been done in regards of anomaly detection using machine learning but the process to develop the most accurate model is still undergoing to this date.

Chapter 2

Related Works

If we go by the fact then it is so surprising to know that almost 2.5 quintillion bytes of data are produced by humans every day. Which brings several challenges when it comes to analyze that amount of dataset manually. In this modern era, almost every tasks are automated and requires a very limited human efforts. Which brings us to another major issue which is the presence of anomalies in the dataset. Technically, it is impossible to keep track of anomalies without any automation being involved. Several researchers are already working on different techniques and algorithms to detect the anomaly in different kind of datasets. The mains issues which are currently being explored are to minimize the imbalanced dataset, data poisoning, real-time detection techniques, selection of model based on the characteristics of the dataset, etc.

2.1 Model Selection Technique

In a recent study, [1] A decision tree or a particular framework is created which helps in determining the model selection suitable for a particular dataset to perform anomaly detection on it. They reviewed the MacroBase architecture and functionality, created benchmark on several commonly used anomaly detection dataset to finally form a decision tree which evaluates the dataset on several factors and gives the best model or algorithm suitable for that dataset.

2.2 Numenta Anomaly Benchmark

In a research paper published by the official Numenta team [3] they introduced the "Numenta Anomaly Benchmark (NAB)" that is basically an open source database and tools which can be used to test and evaluate anomaly detection algorithms on streaming data. The NAB uses a benchmark dataset with labeled real-world time-series dataset which helps in identifying the score and accuracy of the most commonly used anomaly detection algorithms. Numenta developed an algorithm based on Hierarchical Temporal Memory, it is a type of continuous learning system which is extracted from the theory of the neocortex [4] and is compatible for the real-time applications.

2.3 Credit Card Fraud Detection

A comprehensive study of using hybrid model to detect credit card frauds is done by [5], In this paper the main focus was to use AdaBoost over several general algorithms such as SVM, Neural Networks, Decision tree, Naïve Bayes, etc. The metrics used for evaluation was the Matthews Correlation Coefficient. Apart from AdaBoost, they have also used the majority voting methods. To check the robustness of the model, data poisoning techniques such as adding noise to the dataset was used. The results was quite promising showing score of 1 on Adaboost while on 30% of the noise data the majority voting method has given a score of 0.95 which is promising. However, there was no clarification of data being imbalanced, and sampling methods were not provided. The dataset was stacked for modelling which may arise the question of whether the model was over-fitted or not.

2.4 Yahoo S5 Benchmark

[6] used the Yahoo S5 Benchmark dataset, which consist of both original and synthetic data with labeled anomalies classes divided into 1s and 0s. They have used several anomaly detection algorithm for Benchmarking purpose. The studies shows that in comparison of several general anomaly detection algorithms, Simple Online Regression Anomaly Detector (SORAD) is more accurate and successful when it comes to correctly identifying the anomalies as compared to rest of the algorithms. However, SORAD

Algorithm score was not performing well on other benchmarking dataset. For e.g. the same model resulted into a poor score when it was performed over Numenta Dataset, which makes the model highly biased and limited to only Yahoo S5 Benchmark dataset.

2.5 Unsupervised Learning: Autoencoders

[7] In this study an auto encoder was used to train the model in an unsupervised manner to learn the efficient data encodings. The represented encoding part of the dataset for any previous transformation such as PCA can be learned by the autoencoders. The LSTM autoencoders algorithm was used on two different kind of dataset, first being synthetic dataset (Artificial) and second one was the publicly available sound frequency dataset.LSTM layers was used to build an autoencoder which consist of two modules: encoder and decoder. The end result for the artificial data illustrate that the anomaly score of autoencoder is higher than other part of the signal. In case of sound signal frequency the accuracy was around 87% which was a slight improvement as compared to model only using LSTM without any autoencoders.

2.6 Forecasting Method ARIMA

[11] In this study, the network traffic is analysed and ARIMA model is used to classify any possible anomalous value. Moreover, the same model is used for traffic characterization. For evaluation metrics Symmetric mean absolute percentage error (sMAPE) was used.

[13] In this study, a novel anomaly detection technique was developed which can be used for real-time streaming data anomaly detection. The algorithm is based on an on-line sequence memory algorithm known as Hierarchical Temporal Memory (HTM). Both real-time streaming data and static data were used to test the algorithm. Numenta Anomaly Benchmark was one of the dataset the algorithm was tested on. The algorithm performed well on both of the dataset. Specifically, the score for NAB dataset was around 65%.

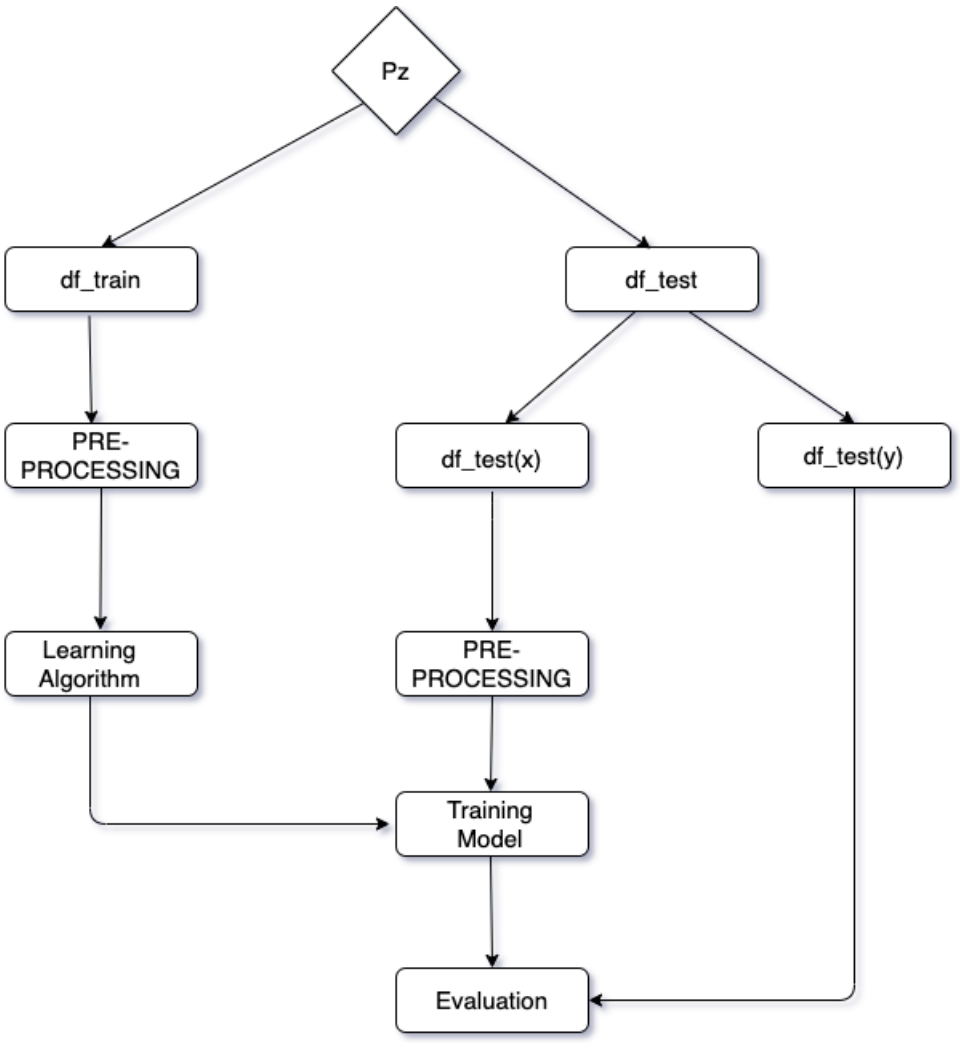
Chapter 3

Methodology

So far we have discussed the summary of our project as well as introduced the related works and studies which were conducted on this domain so far. In this chapter we will discuss the in-depth insights about the methods which was carried out to both extract, process and analyse the dataset which will later be used to deploy a successful anomaly detection model.

3.1 Design

Figure 3.1 illustrate the approach used for processing, analyzing and applying algorithms on the "Yahoo S5 Benchmark Dataset" for predicting the anomalies on both synthetic and original dataset. This dataset contains 371 files namely A1, A2, A3 and A4 benchmark. The dataset has been pre-processed and feature extraction has been performed on it. Afterwards, the target class has been normalized into 0s and 1s. After pre-processing the dataset, several algorithms was used to train the model which was later used for Benchmarking purpose. For testing purpose, the target class was split into two parts 1st being used to evaluation purpose while 2nd part being used for deployment testing phase. The values predicted was then rounded off into two classes 0s and 1s. ARIMA model was used to both predict the values as well as to plot trends and seasonality of the dataset.



- Pz: Original Data Distribution.
- df_train: Training Dataset.
- df_test: Testing Dataset.

Figure 3.1: System Architecture for Yahoo S5 Benchmark Dataset

3.2 Data Management

In this section we are going to discuss and provide all the necessary information about the datasets which were used for this project. This section provides several information about the dataset internal structure such as format of the dataset, features of the dataset, source of the dataset.

3.2.1 Data Collection

In this research, we have used three different datasets to achieve our objective. The following are the datasets which are being used for this research project.

1. S5 - A Labeled Anomaly Detection Dataset, version 1.0: This dataset was released for the researchers by Yahoo Computing Systems Data. The aim of this dataset is to benchmark any anomaly detection algorithms. This dataset is available for anyone who have access to educational email account. It is widely available for download on Yahoo Labs Website.

2. Numenta Anomaly Benchmark (NAB) - The Numenta Anomaly Benchmark (NAB) is a benchmark dataset which is used for evaluating anomaly detection algorithms in any streaming dataset including time-series dataset. It was released by Numenta Corporation (a team of scientists and engineers applying neuroscience principles to machine intelligence research). It was first released in 2015 and was open-sourced on GitHub. The dataset has been downloaded from GitHub, with licence agreement of "GNU AFFERO GENERAL PUBLIC LICENSE" in which we have permission for Commercial use, Modification, Distribution, Patent use and Private use.

3. Credit Card Fraud Detection - The dataset has been collected and analysed during a research collaboration of Machine Learning Group of Université Libre de Bruxelles and the Worldline on big data mining and fraud detection. The research led to several extraction and modification sourced from european bank which is

highly transformed to protect the confidentiality and privacy of customers. The transformed dataset was later made publicly available on Kaggle as a challenge to detect credit card fraud transaction using machine learning. It is available under licence of "Open Database". It can be downloaded from Kaggle or from the Machine Learning Group of Université Libre de Bruxelles(http://mlg.ulb.ac.be)

3.2.2 Dataset

S5 - A Labeled Anomaly Detection Dataset, version 1.0

This dataset contains 371 files namely A1, A2, A3 and A4 benchmark. This dataset can be used to benchmark any anomaly detection algorithm. It includes real and synthetic Yahoo! production traffic. Released early in 2015, the data set brands itself as being the first of its kind to be a data stream data set with labeled anomalies.
This data set includes four subsets of the data, the first being the A1Benchmark. This subset of data includes 67 files of real data where each timestamp contains a value specific to the amount of traffic on that site for the interval of one hour. Each of these data points is then labeled whether it is or is not an anomaly. Each data point in the A1Benchmark data was labeled an anomaly by a human being, so the authors of the data set warn that this data may be best suited to measure recall since the labeling of the anomalies may be subjective to the labelers of the data.
The next of the four subsets is the A2Benchmark data, which consists of the same formatting as the A1Benchmark data, but now the data is synthetic. Each data point in the time-series of the 100 different test files now has a random seasonality, trend, and noise, with anomaly inserted randomly into data set. The A3Benchmark data set only contains anomaly in its 100 files, excluding change-points that would be classified as anomalies. The data attributes have varying noise and trend within the data, and like previous data set, include anomalies at random positions in the data. Finally, data attributes have varying noise and trend in the A4-Benchmark data set. However, change-point anomalies have been included randomly as anomaly in addition to the other types of anomaly in the 100 testing data sets. The anomalies in A1-Benchmark are marked by humans and therefore may not be consistent, therefore during benchmarking A1-Benchmark is best used to measure recall. The fields are: 0 timestamp 1 value 2 is-anomaly. The is-anomaly field is a Boolean indicating if the current value at

a given timestamp is considered an anomaly.

TYPE	DATASET	Observation				Anomalies			
		Min	Mean	Max	Total	Min	Mean	Max	Total
Synthetic	33	1421	1594	1680	52591	1	4.03	8	133
Real	67	741	1415	1461	94866	0	2.13	5	143
Total	100	741	1475	1680	147457	0	2.76	8	276

Table 3.1: Description of Yahoo S5 Benchmark Corpus

As illustrated in Figure 3.2 A typical subset of the dataset has been visualized to see the trends, it can been seen that a sudden spike in the values in between the date of 2014-12-20 to 2014-12-24 has occurred the same can been seen on the 30^{th} of December 2014. The trends are almost similar on different subsets of the dataset.

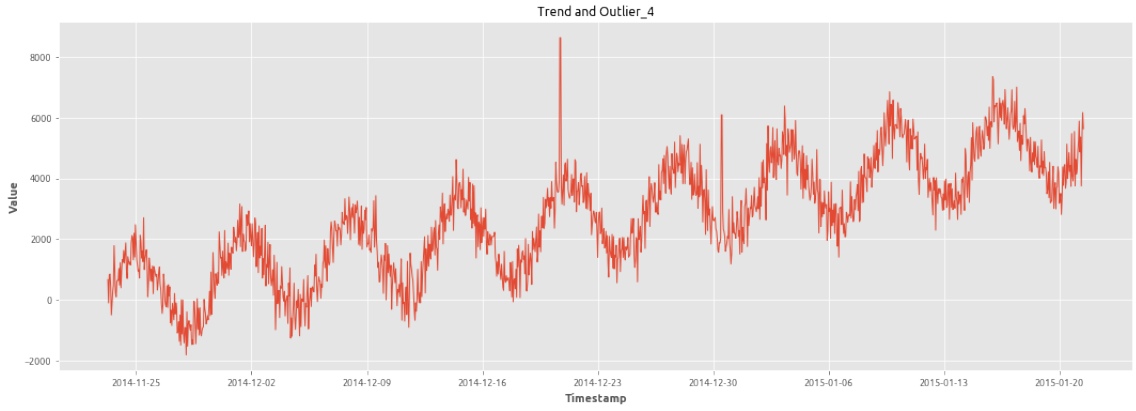


Figure 3.2: Visualizing A1-Benchmark Trend and Outliers.

Numenta Anomaly Benchmark (NAB)

The Numenta Anomaly Benchmark (NAB) is a benchmark dataset which is used for evaluating anomaly detection algorithms in any streaming dataset including time-series dataset. This dataset contains both real-world and artificial time-series data-points. Per series contains around 1000-22000 observations. It also consist of a new type of scoring mechanism specially designed for real-time applications. The data are in ordered manner with timestamped assigned to the values. The real-world data has been extracted from several sources such as AWS server metrics, advertisement clicking metrics, traffic data, Twitter volume, etc.

TYPE	**DATASET**	**Observation**				**Anomalies**			
		Min	**Mean**	**Max**	**Total**	**Min**	**Mean**	**Max**	**Total**
Synthetic	11	4032	4032	4032	44352	0	0.55	1	6
Real	47	1127	6834	22695	321206	0	2.43	5	114
Total	58	1127	6302	22695	365558	0	2.07	5	120

Table 3.2: Description of Numenta Anomaly Benchmark Corpus

In the Figure 3.3, A typical subset of the dataset has been visualized to see the trends. The dataset used here is taken from the "ambient temperature system failure" which is a real-world dataset. It contains the timestamp and the values corresponding to it. A few spikes in the values as well as values getting ceased for a long time can be seen.

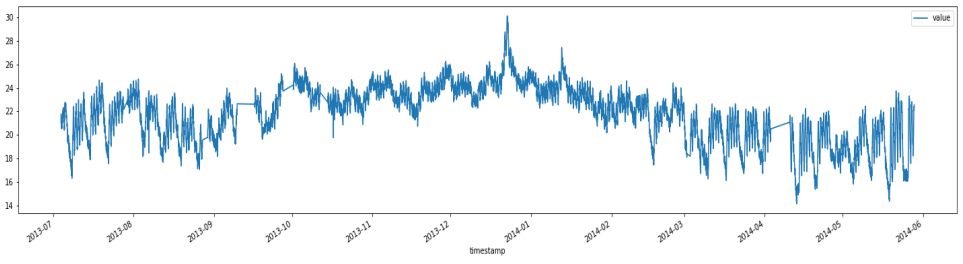


Figure 3.3: Visualizing an Example of time-series dataset from NAB corpus.

Credit Card Fraud Detection

This dataset contains all the electronic transactions made by credit cards in the month of September 2013 by the european cardholders. This dataset is limited to credit card issued by banks inside the european union only. It contains all the transactions that occurred within the time frame of 2 days, in which we have 492 frauds out of 284,807 transactions. The nature of this dataset is highly unbalanced. The dataset contains only numerical values as a result of undergone PCA transformation of all the features to protect the confidentiality and privacy of customers data. It contains Features labelled as V1, V2, ... V28 which are the principle components obtained from the PCA transformation, except two features named "Time" and "Amount" which is in thier original format without any transformation. Feature "Time" contains the seconds elapsed between each transactions and the first transaction. [14]

Measures	Time	V1	V2	V3	V26	V27	V28	Amount	Class
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	284807.000000	284807.000000
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	1.699104e-15	-3.660161e-16	-1.206049e-16	88.349619	0.001727
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	4.822270e-01	4.036325e-01	3.300833e-01	250.120109	0.041527
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-2.604551e+00	-2.256568e+01	-1.543008e+01	0.000000	0.000000
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-3.269839e-01	-7.083953e-02	-5.295979e-02	5.600000	0.000000
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-5.213911e-02	1.342146e-03	1.124383e-02	22.000000	0.000000
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	2.409522e-01	9.104512e-02	7.827995e-02	77.165000	0.000000
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	3.517346e+00	3.161220e+01	3.384781e+01	25691.160000	1.000000

Table 3.3: Description of Credit Card Fraud Transaction Dataset.

In Figure 3.7 The Features for E.g. V1,V2,..,V28 are plotted on Y axis and the time of the transactions are plotted on X axis. The blue color represents the class 0 which means that it is tagged as normal transaction, on the other hand the orange color represents the class 1 which means that the transaction is tagged as a fraudulent transaction. As we can interpret from the figure that there is not a strong co-relation between the timing of transaction and the transaction being fraud or not.

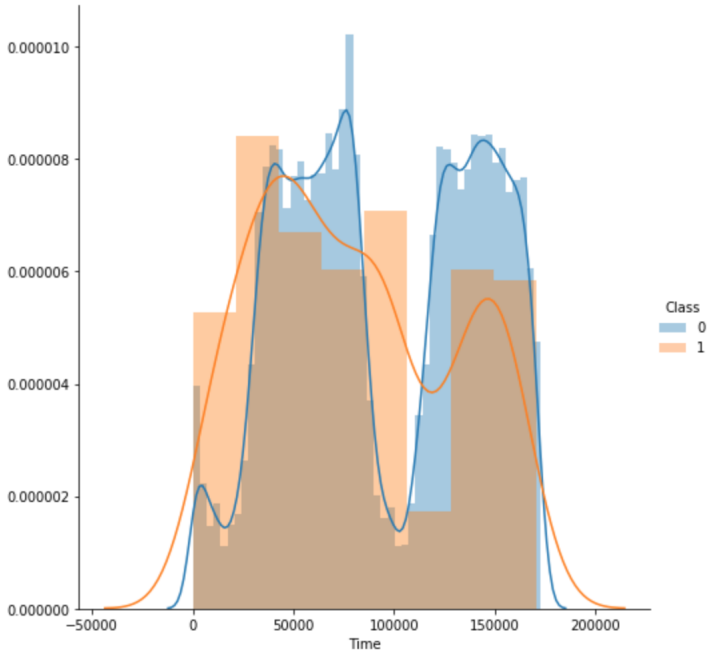


Figure 3.4: Features on Y-axis and Time of Transaction on X-axis.

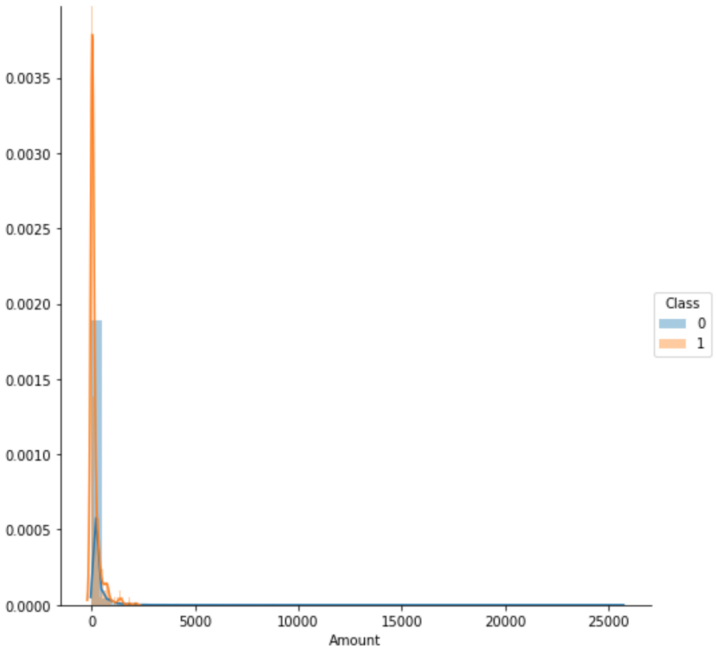


Figure 3.5: Most of the transactions are below 2000 euro for both fraud and normal transactions

3.3 Data Pre-Processing

3.3.1 S5 - A Labeled Anomaly Detection Dataset, version 1.0

Merging Data

The S5 Anomaly detection data contains several subsets which is then divided into further comma separated value files. For e.g A1 Benchmark contains 67 individual files. The first task we performed was to merge the dataset into a single dataframe which will later be used to perform further transformation on it before moving forward to using it to train our model.

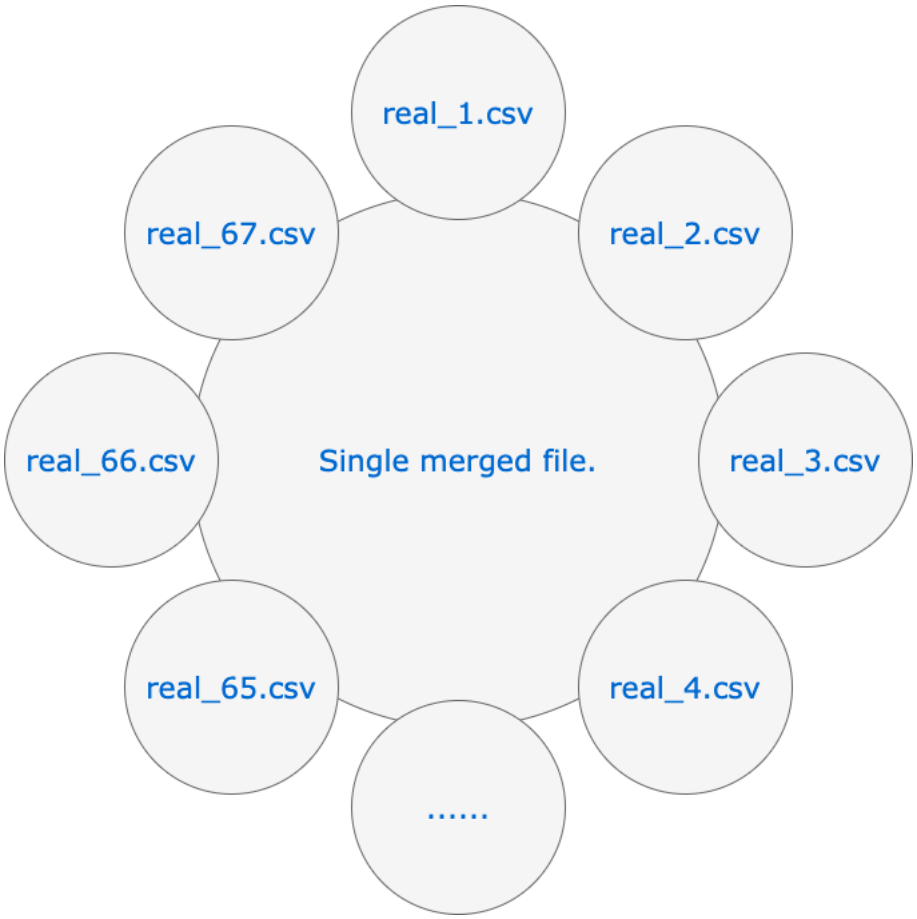


Figure 3.6: Features on Y-axis and Time of Transaction on X-axis.

Splitting the Dataset.

After performing the merging operation, the dataset was then divided into two parts one being used for training purpose while other being used for the testing purpose. This process was achieved using the train-test split method provided by sklear model selection library.

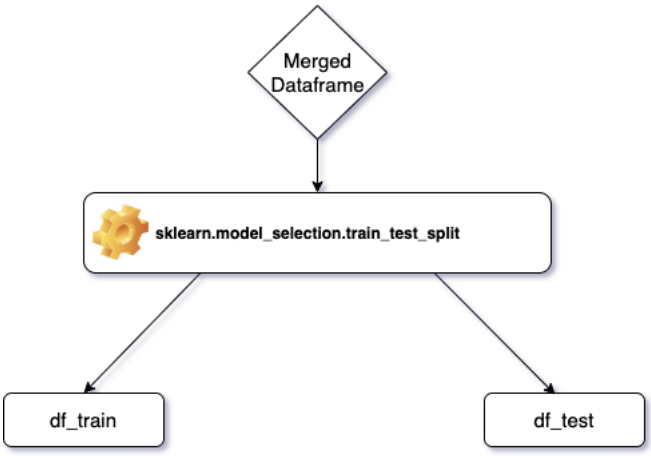


Figure 3.7: Using sklearn library to split the dataset.

Normalisation

Normalisation is a process of rescaling the data from the original range and to convert it into the range of 0s and 1. The values in the dataset are spread into hundreds or thousands of units, which makes the model training difficult to map all these values correctly, or it can become unstable and will be highly senstive to higher values making the prediction highly biased. Therefore, to scale all the features into a simpler way to make it easier for model to train upon it, we first used Min-Max Scaler. In Min-Max Scaler it simply means that the minimum and maximum value of a feature or variable is going to be 0 and 1, respectively. The mathematical formula for the same is:

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Unfortunately, resulting to this our model became highly biased because of the outliers present in the dataset. However, instead of using Min-Max Scaler and switching to RobustScaler helped in normalising the dataset as well as keeping the outliers in mind

it didn't affect the model integrity and performance. RobustScaler scale the features but it also uses statistics that are robust to outliers. The scaler perform by removing the median and then scale the values according to the quantile range. The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

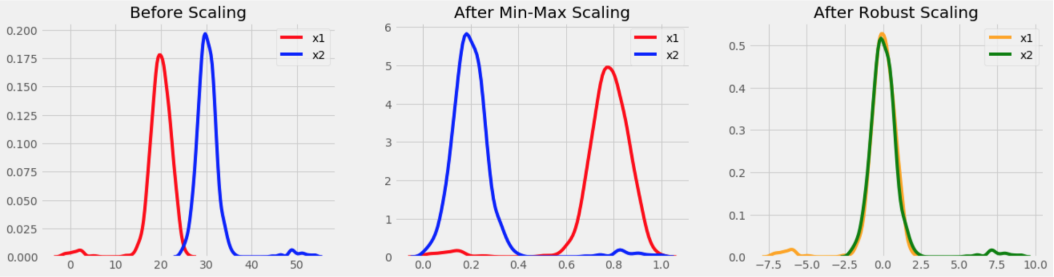


Figure 3.8: Comparison of different normalisation.

3.3.2 Numenta Anomaly Benchmark (NAB)

Feature Engineering

To understand the time-series trends and pattern in the dataset, some modification and feature engineering was performed. The hours was extracted from the time-series and was later converted into daylight or night-time and also a weekdays and weekend pattern was generated. In total 4 categories was generated. The Figure 3.8 below shows the temperature during a particular time of the day, the dataframe which was used is the ambient temperature system failure dataset.

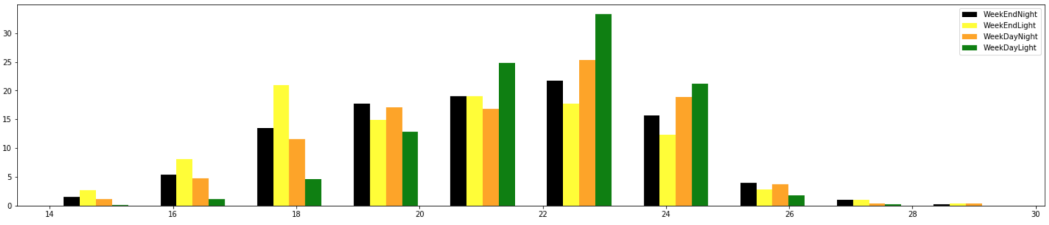


Figure 3.9: Temperature is more stable during daylight of business day.

3.3.3 Credit Card Fraud Detection

The dataset has been thoroughly checked and there is no sign of missing values. Hence, there is no need to perform any missing value operation on it. However, some basic EDA was performed before moving to any further steps.

Exploratory Data Analysis

In Figure 3.10 below indicate the number of class labelled as either 0 or 1, where 0 indicate that the transaction is normal and on the other side 1 indicates that the transaction is tagged as abnormal transaction (Fraud transaction). Total of 286142 transactions are tagged as normal transaction and only 492 transactions have been labelled as fraudulent activity, which makes the dataset highly skewed.

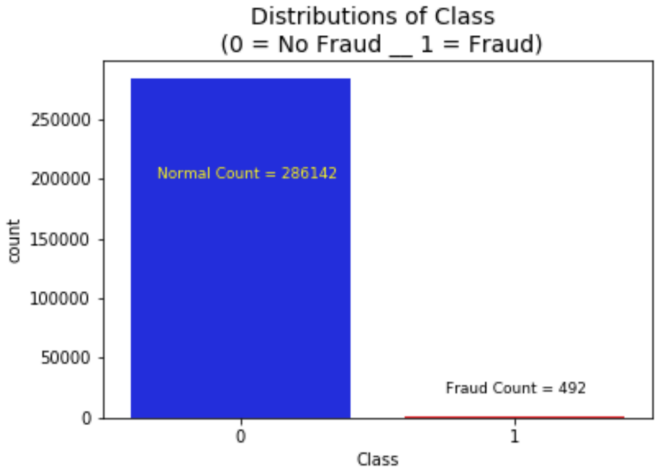


Figure 3.10: Highly Imbalanced Class Distribution.

By Looking at the Figure 3.11 it can be said that all of the fraud transactions are below 2500 euros in which 80% of them are below 500 euros.

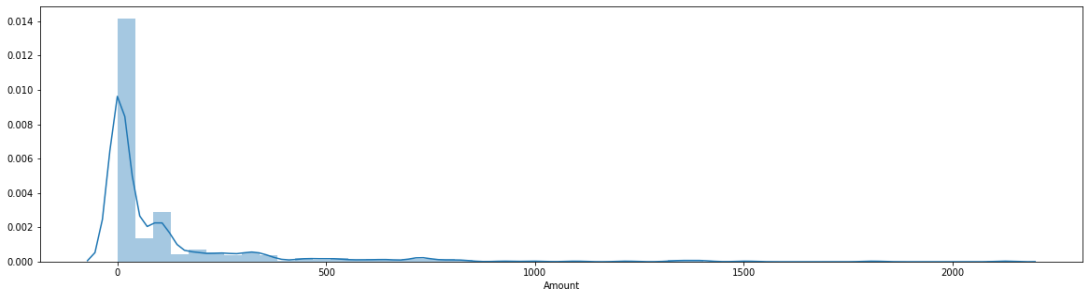


Figure 3.11: Fraud Transactions Amount.

Figure 3.12 illustrate the distribution of transactions over time, it can be seen that the distribution is widely spread and it is important to scale them to make the training process both easier and less sensitive to the values, we created a sub-sample of the data, so that our model can understand the pattern and understand whether the transaction falls under fraud transaction or not. It is a key part of our research to balance the ratio of fraud and normal transaction so that our model can predict more accurately without being biased and over-fitted. The problem with the dataset is that there is only 492 fraudulent transaction which makes the training process difficult to properly understand the trend and factors responsible for fraudulent activity.

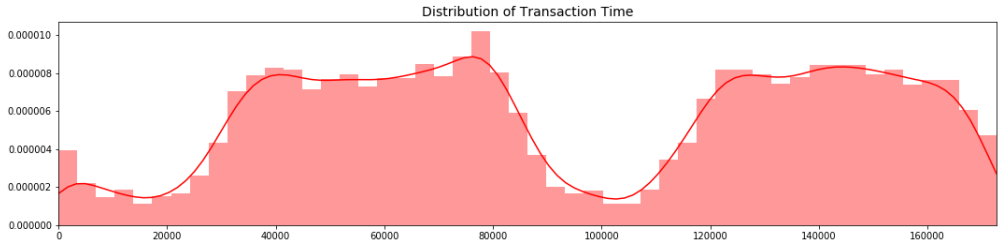


Figure 3.12: Distribution of Transaction Time.

To balance we have used the following algorithms.

- Random Under-Sampling
- Over-Sampling Technique

But before performing any sampling over the dataset we need to split the dataset for our testing purpose. We have used the sklearn model selection train-test split method to split the dataset into 70-30 ratio of train test subsets.

```
No Frauds 99.83 % of the dataset
Frauds 0.17 % of the dataset
Train: [ 30473  30496  31002 ... 284804 284805 284806] Test: [    0     1     2 ... 57017 57018 57019]
Train: [     0      1      2 ... 284804 284805 284806] Test: [ 30473  30496  31002 ... 113964 113965 113966]
Train: [     0      1      2 ... 284804 284805 284806] Test: [ 81609  82400  83053 ... 170946 170947 170948]
Train: [     0      1      2 ... 284804 284805 284806] Test: [150654 150660 150661 ... 227866 227867 227868]
Train: [     0      1      2 ... 227866 227867 227868] Test: [212516 212644 213092 ... 284804 284805 284806]
---------------------------------------------------------------------------------------------------
Label Distributions:

[0.99827076 0.00172924]
[0.99827952 0.00172048]
```

Figure 3.13: Distribution of the labels

Random Under-Sampling

Random Under-Sampling basically means that we remove some part of the data in order to have a more balanced dataset which contains equal amount of target labels, this process helps to keep the model away from overfitting. As discussed in Figure 3.10 we can see that we have around 492 fraud cases, so we had to bring it to 50-50 ratio to make the data balanced, doing so resulted into same amount of non-fraud cases too (e.g 492 non-fraud cases). The only risk we are taking here is that in this whole process we are sacrificing a big amount of data which are being deleted to make the data balance, which will make our model a less accurate.

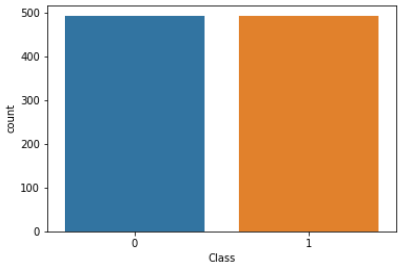


Figure 3.14: Equally Distributed Classes

Synthetic Minority Over-Sampling Technique (SMOTE)

When using machine learning models, it can be difficult to maintain a good score when one class in training dataset dominates the other. Our dataset is highly imbalanced (E.g. very high number of normal transaction as compared to fraud transaction). To solve this particular problem of data being imbalanced, we use Synthetic Minority Over-Sampling Technique. In SMOTE, the minority class (in our case fraud transaction) is increased to level up with the majority class (in our case normal transaction). New synthetic dataset is generated which maintains a good ratio of minority and majority class to balance the dataset. It randomly picks a point from the minority class and compute the K-nearest neighbors for that particular point. Then the synthetic points are added between its neighbours and the chosen point.

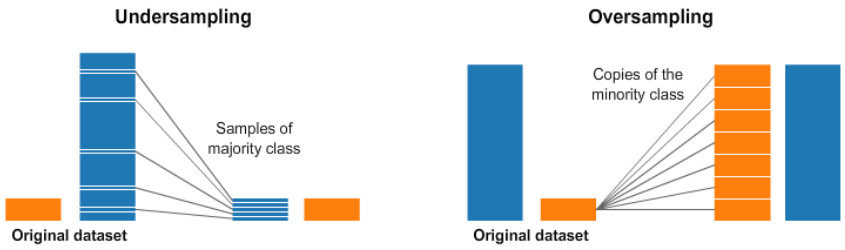


Figure 3.15: Difference between Under-Sampling and Over-Sampling.

Correlation Matrix

Correlation Matrix plays a very important role when we are trying to understand our data. The first thing which we want to know is that if we have any features available which influence heavily in whether a transaction is fraud or not. However, we need to keep this in mind that using the correlation matrix on the original data can't be used as a reference because of the imbalance nature of it. So we need to make sure that we use the sub-sample dataset correlation matrix to interpret any outcomes.

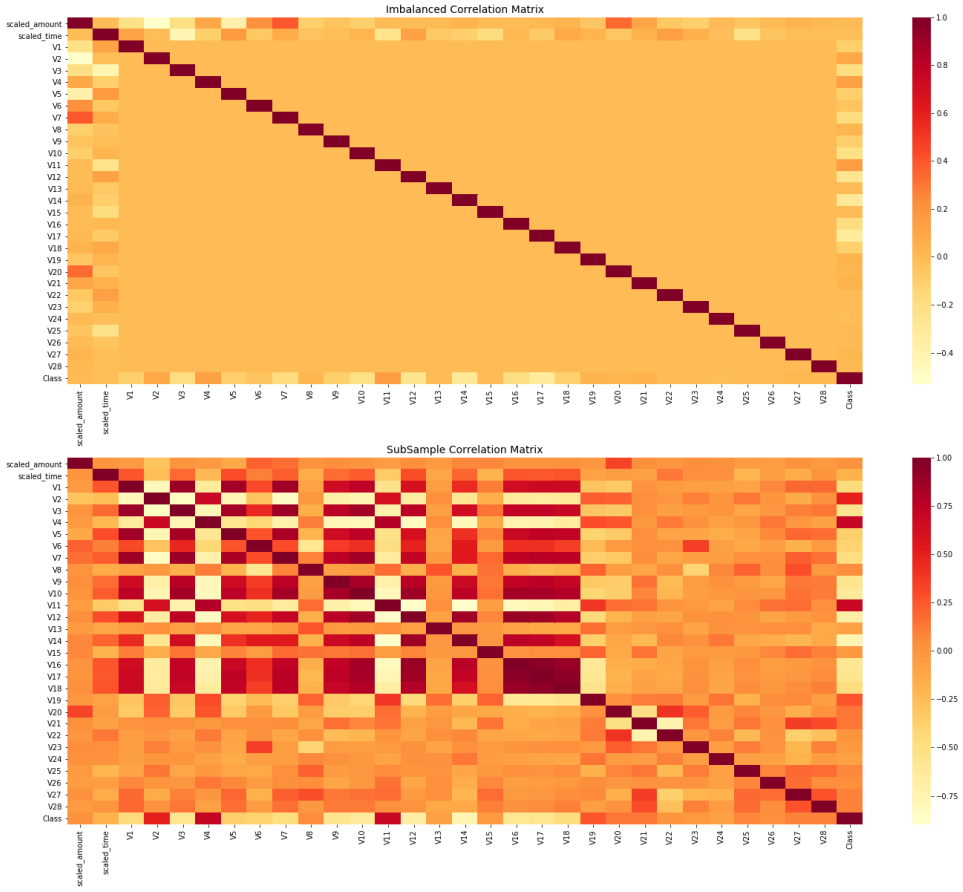


Figure 3.16: Correlation Matrix Comparison Before and After Under-Sampling.

Correlation matrix consist of two different types of correlation: positive correlation and negative correlation. Here V-10, V-12, V-14 and V-17 are negatively correlated. Which means that the lower these values get the higher the chance of target is to be a fraud transaction. V-2, V-4, V-11, and V-19 are positively correlated. Higher values are more likely to be classified as fraud transaction.

Box Plot Analysis

Box Plot is a way to summarize a set of data which is measured on an interval scale. To better understand the distribution of these features in fraudulent and non fraudulent transactions we used Box Plots.

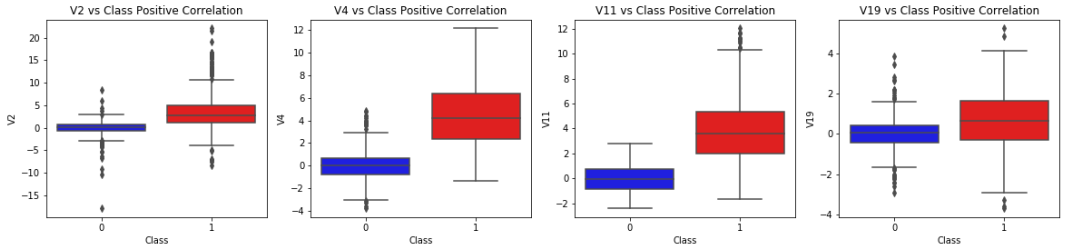


Figure 3.17: Positive Correlation Box Plot Analysis.

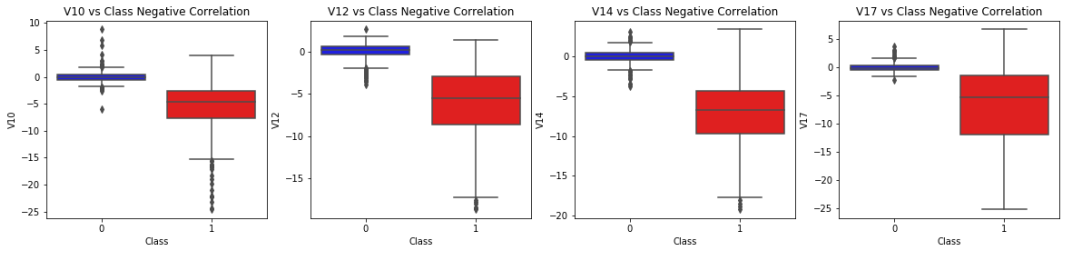


Figure 3.18: Negative Correlation Box Plot Analysis.

Outlier Removal Tradeoff

Any extreme outliers from the features that have high correlation with the classes have been removed. It will help in improving the model accuracy. In doing so we have to be extra caution to keep in mind how far do we need to set the threshold for removing outliers. The threshold has been determined by multiplying a number by the IQR. Higher threshold means less outlier detection and vise versa.

After determining the number which we used to multiply with the IQR, we then proceed to determine the upper and lower thresholds by subtracting $q25 - threshold$ and $q75 + threshold$

```
Quartile 25: -9.692722964972385 | Quartile 75: -4.282820849486866
iqr: 5.409902115485519
Cut Off: 8.114853173228278
V14 Lower: -17.807576138200663
V14 Upper: 3.8320323237414122
Feature V14 Outliers for Fraud Cases: 4
V10 outliers:[-18.4937733551053, -19.2143254902614, -18.8220867423816, -18.049997689859396]
----------------------------------------------------------------------------------------------------------------------------------
----------
V12 Lower: -17.3430371579634
V12 Upper: 5.776973384895937
V12 outliers: [-18.4311310279993, -18.553697009645802, -18.683714633344298, -18.047596570821604]
Feature V12 Outliers for Fraud Cases: 4
Number of Instances after outliers removal: 975
----------------------------------------------------------------------------------------------------------------------------------
----------
V10 Lower: -14.89885463232024
V10 Upper: 4.920334958342141
V10 outliers: [-15.2399619587112, -24.5882624372475, -15.563791338730098, -24.403184969972802, -16.6011969664137, -2
2.1870885620007, -15.2318333653018, -15.2399619587112, -20.949191554361104, -14.9246547735487, -18.2711681738888, -1
5.563791338730098, -14.9246547735487, -16.3035376590131, -16.2556117491401, -19.836148851696, -15.1237521803455, -15.
124162814494698, -22.1870885620007, -17.141513641289198, -22.1870885620007, -18.9132433348732, -15.346098846877501, -
16.7460441053944, -16.6496281595399, -22.1870885620007, -23.2282548357516]
Feature V10 Outliers for Fraud Cases: 27
Number of Instances after outliers removal: 947
```

Figure 3.19: Removing Outliers (Highest Negative Correlated with Labels)

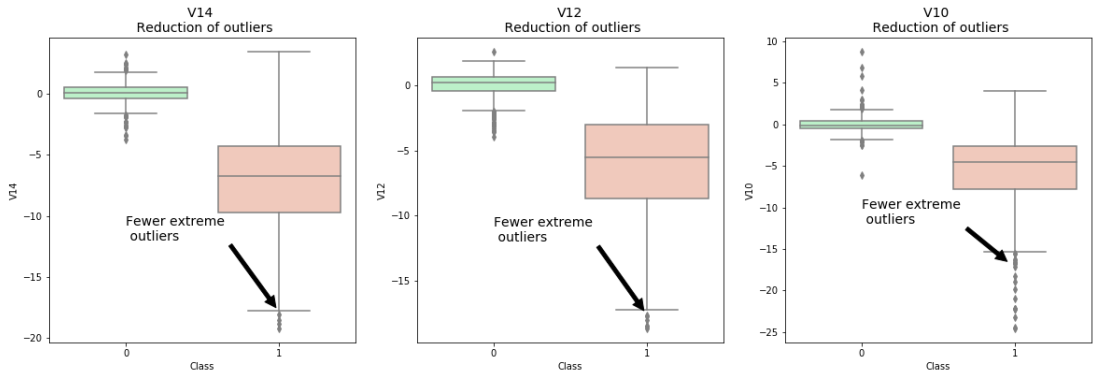


Figure 3.20: Removing Outliers (Boxplots with outliers removed)

Dimensionality Reduction

For this process, we used t-SNE and Principal Component Analysis(PCA) were used to both cluster the target class and reduce the dimensions of the dataset respectively. PCA was able to reduce the dimension of the dataset. After performing PCA transformation, t-SNE was able to accurately cluster the cases fraud and non-fraud in the dataset. More about PCA and t-SNE algorithms is explained in section 4.3

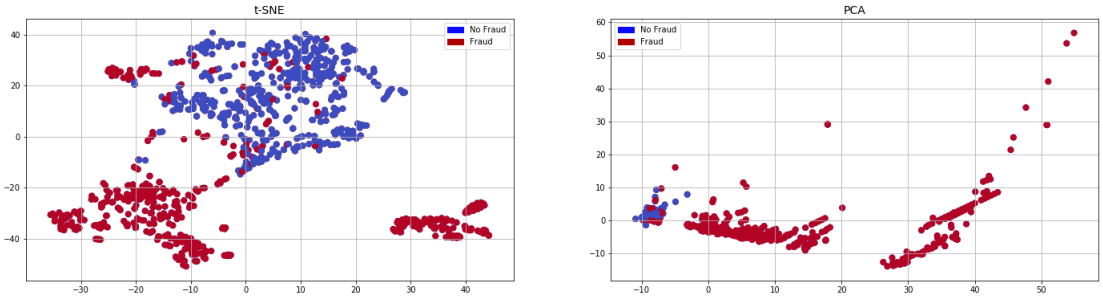


Figure 3.21: Clusters using Dimensionality Reduction)

Chapter 4

Models and Algorithms

In chapter 4 we are going to discuss the in-depth description of the methods and algorithms which were carried out to to perform the task. We are going to cover a brief description of the mathematics and logic behind the algorithms which were used to both prepare and train our dataset. We are going to divide the sections based on the datasets and the work done on it.

4.1 S5 - A Labeled Anomaly Detection Dataset, version 1.0

4.1.1 Autoregressive Integrated Moving Average (ARIMA)

ARIMA model which is short for "**A**utoregressivev **I**ntegrated **M**oving **A**verage is divided into different class of models that forecast a given time-series data based on its own past values, that can be it's own lags and lagged errors in the forecast, and then uses that information to forecast future values. An ARIMA model can be characterised by 3 terms namely:

- p: order of the AR term (lag order)
- d: n^{times} the raw order are differenced (differencing degree)
- q: order of the MA term (moving average order)

1. **AR (Autoregression):** The term "Auto Regressive" means that it is a type of linear regression model that uses self lags as predictors. It is not dependent on any variables. It can be said that Auto Regressive is a function X_t which is a "lags of X_t"

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t$$

2. **I (Integrated):** This model is used for differencing the raw observation. For e.g if there is an observation x_i and the previous time step observation was x_k it then subtract like the following $x_i - x_k$. In statistics differencing is used to transform time-series data to a stationary value. This makes the property independent and it is no longer dependent on the time of observation. Moreover, it eliminates any possible trends and seasonality of the data which makes the mean of time-series stable.

$$\begin{aligned} y_t^* &= y_t' - y_{t-1}' \\ &= (y_t - y_{t-1}) - (y_{t-1} - y_{t-2}) \\ &= y_t - 2y_{t-1} + y_{t-2} \end{aligned}$$

3. **MA (Moving Average):** This model is where the X_t depends only on the lagged forecast errors. This model uses the support between an observation and the residual error from MA models which is applied to lagged observations. Moving Average model is restricted to always being stationary. In the formula below q = number of lagged observation to take in.

$$X_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$$

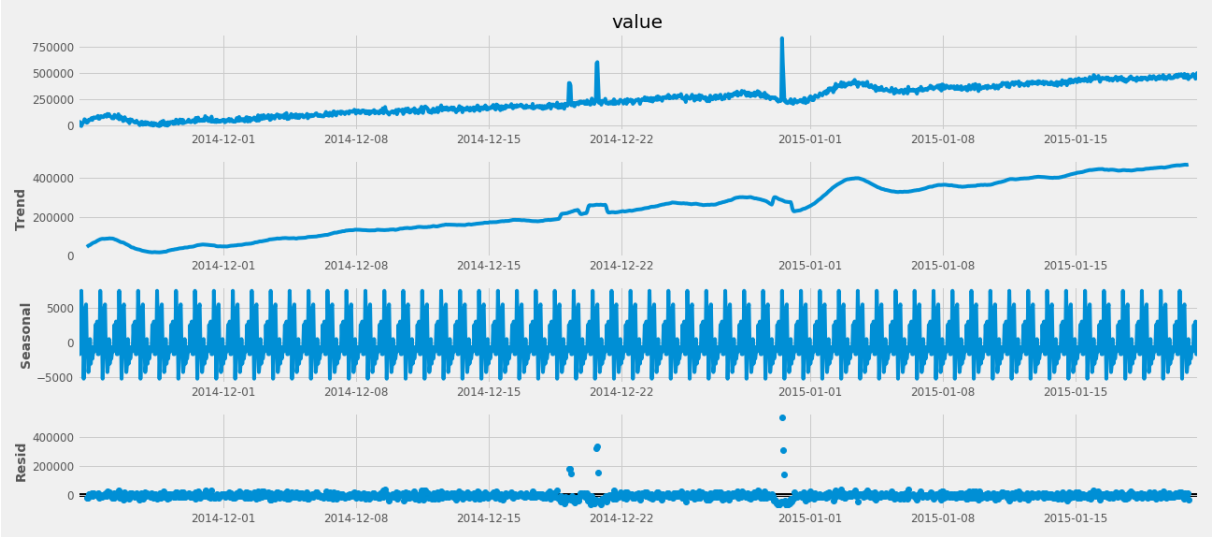


Figure 4.1: Trends and Seasonality of Data

4.1.2 OLS and Ridge Regression

OLS Regression short for "Ordinary least squares" which comes under the family of linear regression, a well known part of linear least squares method. It is used to predict the unknown parameters in a linear regression model. If we talk about Simple linear regression then, it is for minimizing the squared errors. But in our case, we don't want to compensate our positive errors by the negative errors.

$$\hat{\alpha} = \min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2 = \min_{\alpha} \sum_{i=1}^{n} \varepsilon_i^2$$
$$\beta = \min_{\beta} \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2 = \min_{\beta} \sum_{i=1}^{n} \varepsilon_i^2$$

- parameter α represents the value of dependent variable.
- parameter β represents dependent variable.
- ε_i = error term.

Ridge Regression: At this stage we want to reduce the model complexity without

sacrificing the model accuracy. By reducing complexity we simply want to reduce the number of predictors. To do so, it can be achieved by setting the model coefficients to zero. Instead of forcing the model to apply the value to exactly zero, we penalize them if they are farther from zero, which makes them to be restricted within a small continuous way. By doing so, we reduce the complexity of the model without loosing any variables in the model. In Ridge Regression, OLS loss function is extended in such way that apart from minimizing the sum of squared residuals, the size of parameter estimates is penalized, in order to reduce them towards 0.

4.1.3 Gaussian Distribution for Anomaly Detection

In Gaussian Distribution approach, we model all the features on a gaussian distribution and new data points are given, in which the probability of the data-points are dependent on the function of the Gaussian/Normal Distribution. If the probability is below the given threshold then the new point is classified as an outlier/ anomaly. The mathematical formula for the same is:

$$p(x) = p\left(x; \mu, \sigma^2\right) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$p(x)$ = probability of x

Gaussian Distribution Anomaly Detection Algorithm:
m = data points
n = selected features

- Mean parameter for each feature (k = 1 to n) is fit.

$$\mu_k = \frac{1}{m} \sum_{i=1}^{m} x_k^i$$

- Variance parameter for each feature (k = 1 to n) is fit.

$$\sigma_k^2 = \frac{1}{m} \sum_{i=1}^{m} \left(x_k^i - \mu_k\right)^2$$

- By a new data-point, $x = (x_1, x_2, \ldots, x_k), p(x)$ is given by,

$$p(x) = \prod_{k=1}^{n} p\left(x_k; \mu_k, \sigma_k^2\right)$$

- At last, a parameter for threshold, $\mathcal{E}$ is selected such that, if $p(x) < \mathcal{E}$ then X is considered as anomaly, otherwise the value is considered as a normal value.

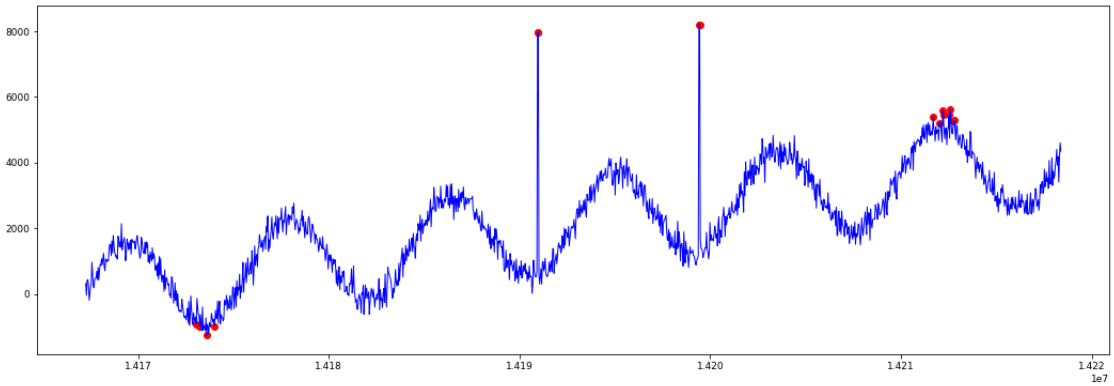


Figure 4.2: visualisation of anomaly throughout time

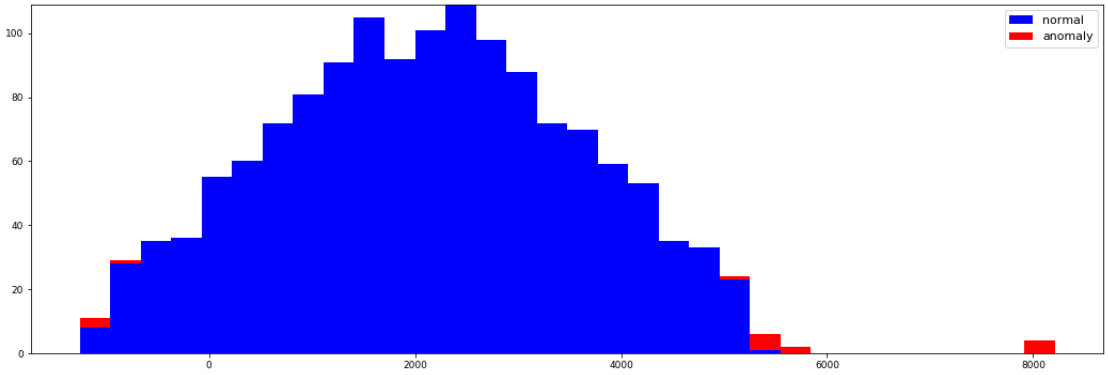


Figure 4.3: visualisation of anomaly with value re-partition.

4.1.4 Recurrent neural network:

RNN is a part of artificial neural network. Here we are using RNN which learn to recognize sequence in the data and then make prediction based on the previous sequence. We consider an anomaly when the next data points vary too much from RNN prediction. Aggregation, size of sequence and size of prediction for anomaly are important parameters to have relevant detection. Here we make learn from 50 previous values, and we predict just the next value. For activation "linear" function is used and loss is calculated using mean-squared-error.

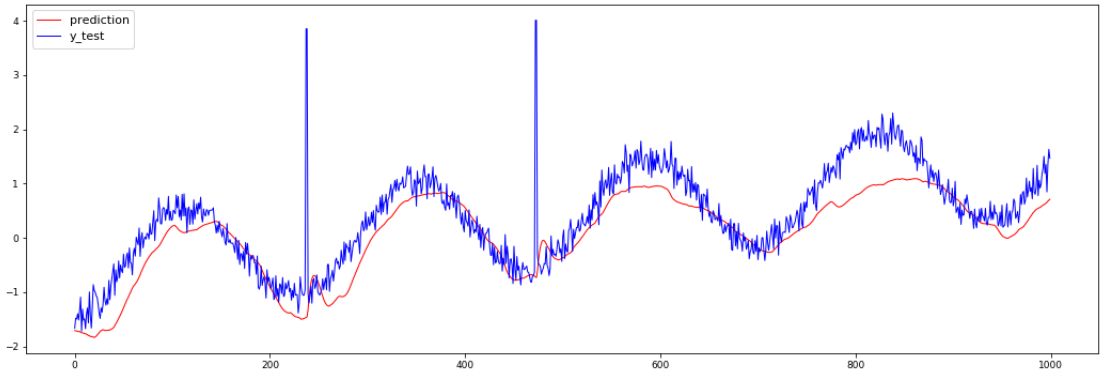


Figure 4.4: Prediction of A2 Benchmark using RNN.

In Figure 4.4 red colour represents the predicted value and blue color is the original value, it can be clearly seen how the prediction trend has no sudden changes whereas, the original data shows that there is two sudden spike in the values, which concludes to the decision that if the distance between the original value and predicted value varies too much then that value is considered to be an anomaly.

4.2 Numenta Anomaly Benchmark

4.2.1 Cluster based anomaly detection (K-mean)

For Numenta data, we used this unsupervised learning technique, this model training involves zero human supervision and no target class is provided. In K-means clustering algorithm, we provide number of clusters, and assign them a colour, that the model will use to group similar type of values together in. K-means will then generate that particular number of centroids which lies at the center of a cluster. We grouped the usual combination of features together. And the points that resulted far from the cluster are considered to an anomalous points. In Figure 4.5 we used elbow method to find the optimal number of centroids.

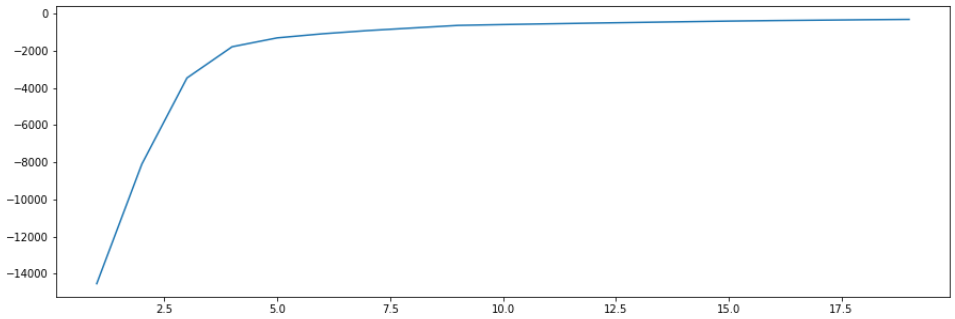


Figure 4.5: Calculation with different number of centroids to see the loss plot.

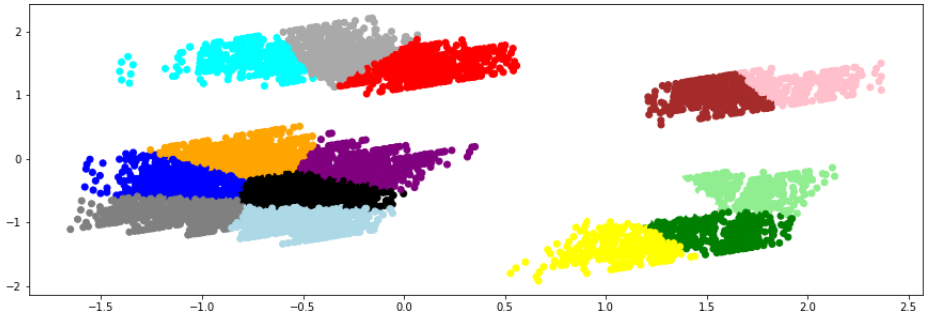


Figure 4.6: plotting the different clusters with the two main features.

We then calculated the distance between each point and its nearest centroid available. The farther distance are then tagged as an anomaly value.

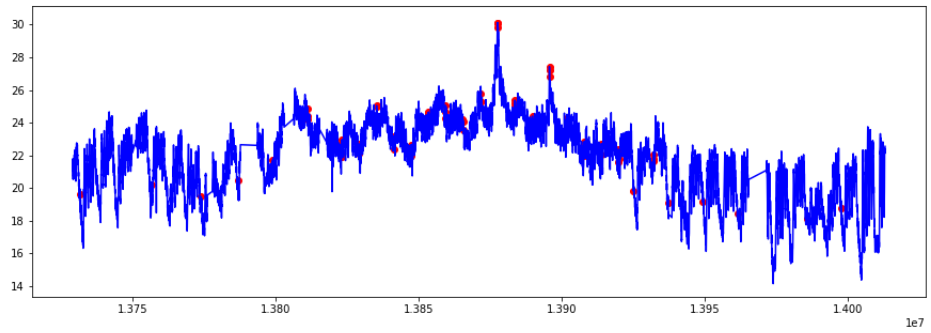


Figure 4.7: visualisation of anomaly throughout time-series.

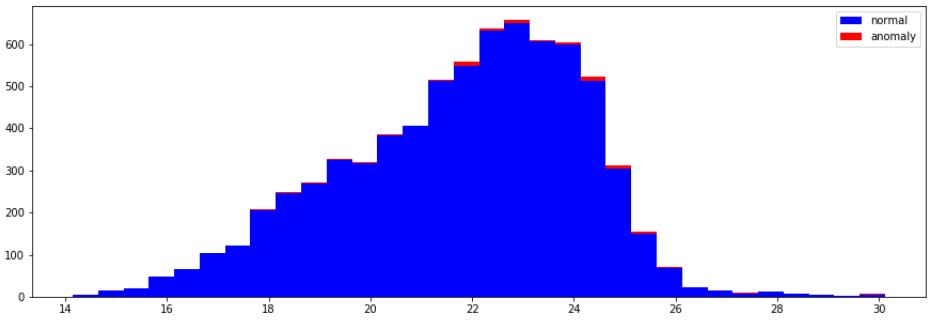


Figure 4.8: Visualisation of anomaly with temperature repartition.

4.2.2 Gaussian Distribution for Anomaly Detection

As discussed above in Section 4.1.3, we are going to test the gaussian distribution method in our Numenta dataset to benchmark the model accuracy, and to see how well it is performing to this dataset. We separated the dataset by what we think of most important categories. Moreover, we also separated the dataset based on the cluster method. Then we moved onto finding outliers (gaussian repartition). We created 4 different dataset based on categories defined before (refer to Figure 3.9). Then we applied elliptic Envelope(gaussian distribution) for each categories. The Elliptical Envelope is used for detecting anomalies in a Gaussian distributed data. A multivariate gaussian distribution is fitted to the dataset. Then a hyperparameter is selected which later will be used as a % of observations the algorithm will assign as an anomalous value.

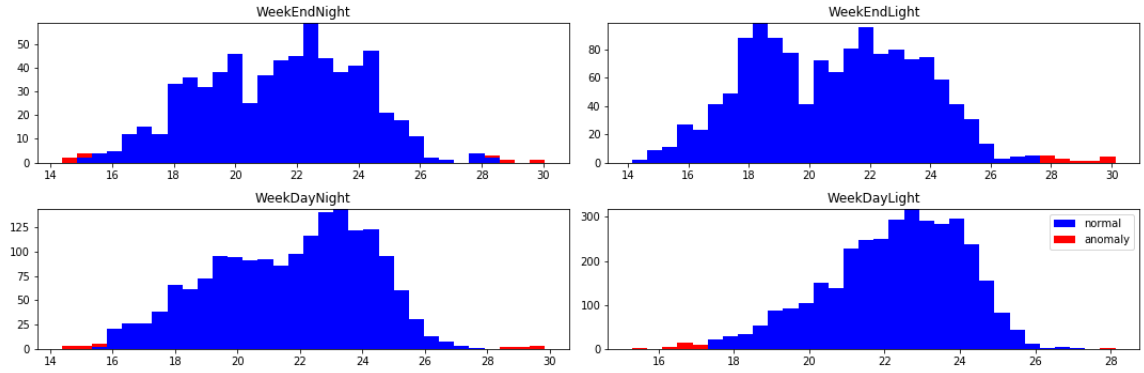


Figure 4.9: Visualisation of the temperature repartition by categories with anomalies.

4.3 Credit Card Fraud Detection

4.3.1 Principal Component Analysis (PCA)

When working on large amount of dataset, the volume of the dataset can easily make any training process long and it requires a huge amount of computational resources. Removing features and entries from the dataset can result into loosing a huge amount of accuracy which can compromise the model integrity. We want to reduce the data complexity and transform it into such a way that does not affect the quality of the dataset. That's where PCA transformation comes into the play. Principal Component Analysis is a dimensionality reduction technique which is used for reducing the dimension of the dataset without loosing valuable infromation from the original dataset. When the PCA transformation is done, a trade-off takes place in which we compromise a little bit of model accuracy for a less complex dataset and to decrease the computation resources the model will take while training our dataset. Steps for any PCA Transformation:

- **Standardization:**
  Here the values of the dataset are standardized into a smaller value, every value of the dataset becomes equally important for the analysis purpose. We first calculate the standard deviation and the mean for each feature available. After that we apply the formula for Standardization:

$$x_{new} = \frac{x - \mu}{\sigma}$$

- **Calculating the covariance matrix:**
  We calculate the correlation matrix to find out the relation between the features and to see if features have any high correlation, if so then one of the feature is dropped which makes the dataset a bit less complex without loosing the relationship between the target class and the features. Then we find out covariance, it gives us the information of the linear relationship between two variables. Information regarding, if the variables are directly or inversely proportional to each other.

The formula for the same is:
For Population:

$$\text{Cov}(x, y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

For Sample:

$$\text{Cov}(x, y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{(N - 1)}$$

As we have already standardized the dataset, which means that the mean of all the feature is 0 and the standard deviation is 1. We then calculate the covariance using the formula above.

- **Eigenvalues and Eigen vectors:**
  Eigenvector is a type of non-zero vector which can carry the most information and when the linear transformation is applied it changes by a scalar factor. Whereas, eigenvalues are the factors given to eigenvectors which determines by by which the eigenvector is scaled or the volume information to be held. This is used for identifying the principle components which was obtained from the linear model of the initial variables.

- **Sorting eigenvalues and their respective eigenvectors.**
  In our case, the eigenvectors are already sorted in ascending order so we just need to decide if we want to keep all of the vectors or remove some of them.

- **Formation of eigenvectors matrix and Transformation of Original Matrix**
  Reshaping the original axes to the new axes derived from the principle axes. It can be achieved by the matrix multiplication of original dataset and the transpose of the feature vectors.

$$Transformed\,Data = FeatureMatrix^T * Standardized\,OriginalData^T$$

OR

$$Transformed\,Data = Feature\,Matrix * top\,k\,eigenvectors$$

4.3.2 t-Distributed Stochastic Neighbor Embedding (tSNE)

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear technique, that is mainly used for dimensionality reduction and to plot the multi-dimensional data onto a two dimensional axis which makes EDA of the data easier to understand and interpret. In simple way, what t-SNE does is if find a way to project data into a low dimensional space so that the clustering in the high dimensional space is preserved. Steps involved in t-SNE algorithm.

- The algorithm first step is to calculate the probability of similarity of each points in both high and low dimensional space.For every point available the gaussian distribution gets calculated. The correlation of the points respective to each other are calculated. After normalizing the points, a set of probabilities for all the points are generated.
- The distance between the conditional probabilities in higher dimensional and lower dimensional is then minimised to form a balanced representation of data-points in lower-dimensional space.
- In the final stage, the variation between the two probability distributions is calculated with the help of Kullback-Leibler divergence, the differences then gets minimised using the gradient descent method.

4.3.3 Logistic Regression

Logistic Regression is a kind of classification model, even though in name the word "regression" is used but instead it uses a "S" shaped curved also known as **Sigmoid** function. The formula of the Sigmoid function is:

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

Logistic regression can be of different types such as:

- **Binary Classification:** This type of logistic regression can only have two outputs, either 0 or 1. For e.g. an email being classified as "Spam" or "Not Spam"

- **Multinomial:** An unordered list of three or more outcomes. For e.g (Disease A, Disease B, Disease C).
- **Ordinal:** Target Variables with ordered categories. For e.g: A customer satisfaction rating ranging categorized as "Very Desatisfied", "Disatisfied", "OK", "Satisfied", "Very Satisfied". Which is then given a score ranging from 1 to 5.

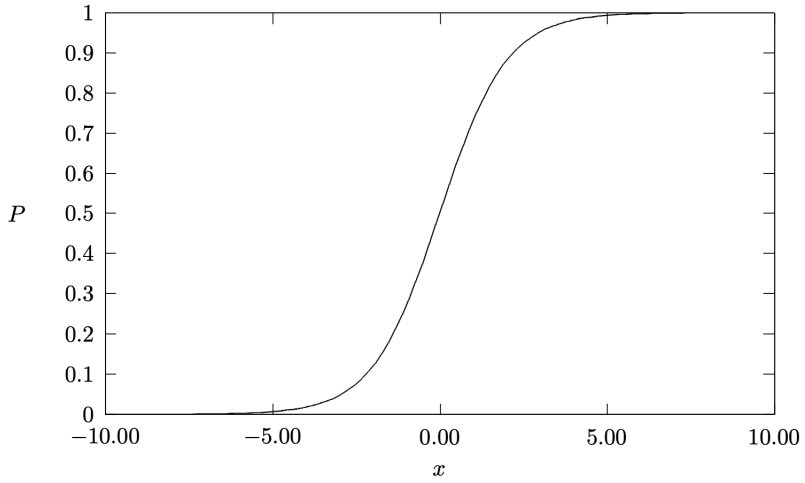


Figure 4.10: Logistic Regression - Sigmoid Function.

4.3.4 Gaussian Distribution for Anomaly Detection

As discussed above in Section 4.1.3, we are going to test the gaussian distribution method in our credit card dataset for benchmarking purpose, The μ and σ for the dataframe variables was calculated which was used in the gaussian function. Then the probability distribution for each row was calculated. Then the epsilon threshold value was selected using the cross-validation technique.

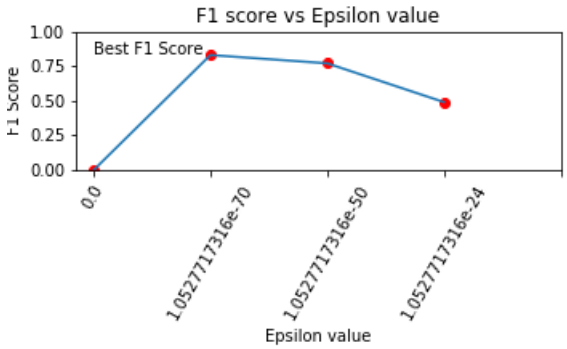


Figure 4.11: Epsilon Crossvalidation (1).

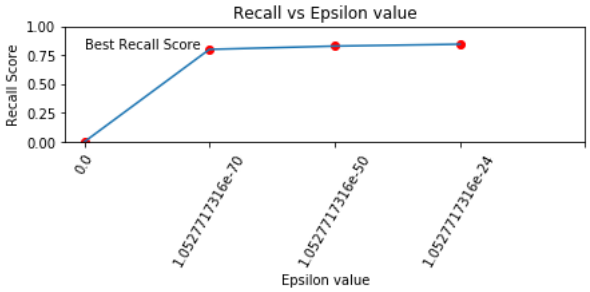


Figure 4.12: Epsilon Crossvalidation (2).

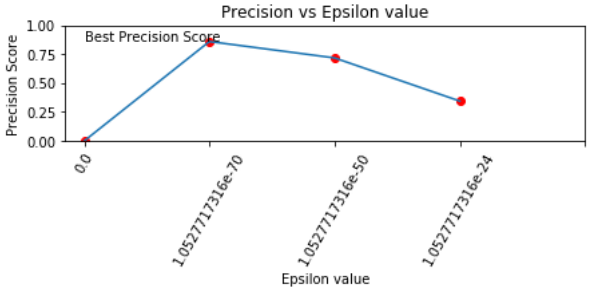


Figure 4.13: Epsilon Crossvalidation (3).

After carefully looking at the cross validation score, Epsilon value = 1.0527717316e-70 was selected as the optimal threshold to detect any anomalous transactions. Then the prediction was done using this threshold and precision, recall and F1 score was calculated. Refer to Table 4.1 for the results.

F1 SCORE	RECALL	PRECISION
0.829474	0.800813	0.860262

Table 4.1: Results on Gaussian Distribution Anomaly Detection

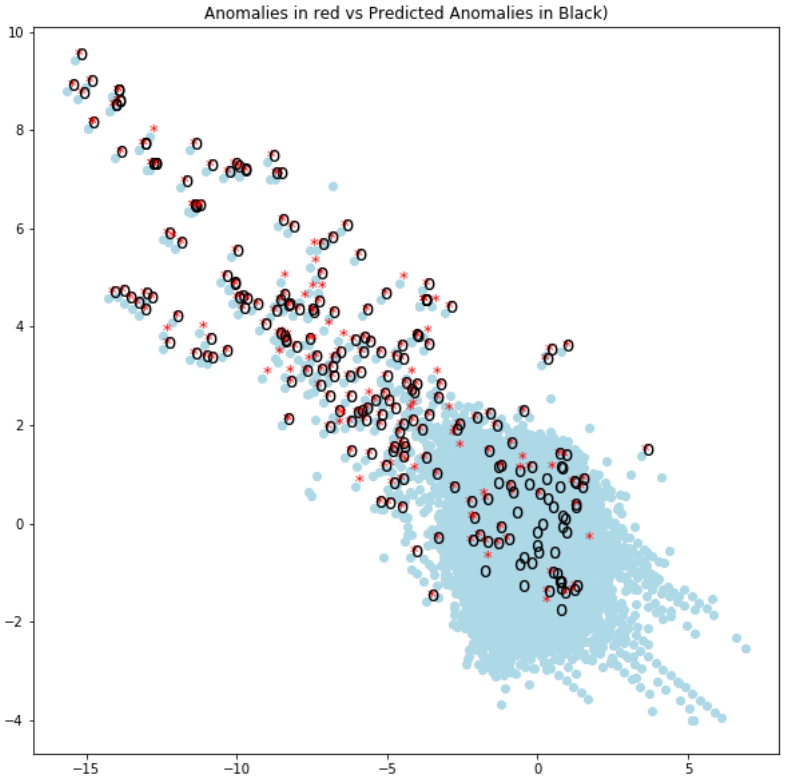


Figure 4.14: Gaussian Distribution Prediction (Credit Card Dataset).

4.4 Evaluation Metrics

When dealing with machine learning models, we split the dataset into two parts, one being training set while other being testing set. This process is done to test our model accuracy and calculate the different evaluation scores such as Confusion matrix, RMSE, MSE, ROC, etc. Each metrics depends on the type of model we are building and which one is more suitable evaluation to perform on certain type of models.

Confusion Matrix

A confusion matrix is a type of N x N matrix, which contains combinations of actual and predicted values. It can be used in classification models. The same metric can also be used to calculate other evaluations such as recall, accuracy, etc. There are mainly four terms which are used in the N x N matrix.

- **True Positive:** It counts number of times when the model predicts the positive class correctly.
- **True Negative:** It counts number of times when the model predicts the negative class correctly.
- **False Positive:** It counts number of times when the model incorrectly predicts the positive class. For e.g. (The email was not a spam but the model classified it as spam).
- **False Negative:** It counts number of times when the model incorrectly predicts the negative class. For e.g. (The email was spam but the model classified it as "Not a Spam").

Precision

Precision gives us the information regarding the proportion of positive identifications which was actually correct. It is a ratio of correctly identified positive predictions made out of the total number of predictions.

$$\text{Precision} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}}$$

Recall

Recall is a ratio of correctly identified positive predictions made out of the total number of positive predictions in the dataset.

$$\text{Recall} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalseNegatives}}$$

F1-Score:

F1 score is the mean value of above two metrics (precision and recall). It gives us value in between 0 and 1, where 0 means worst case and 1 denotes perfect precision and recall.

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

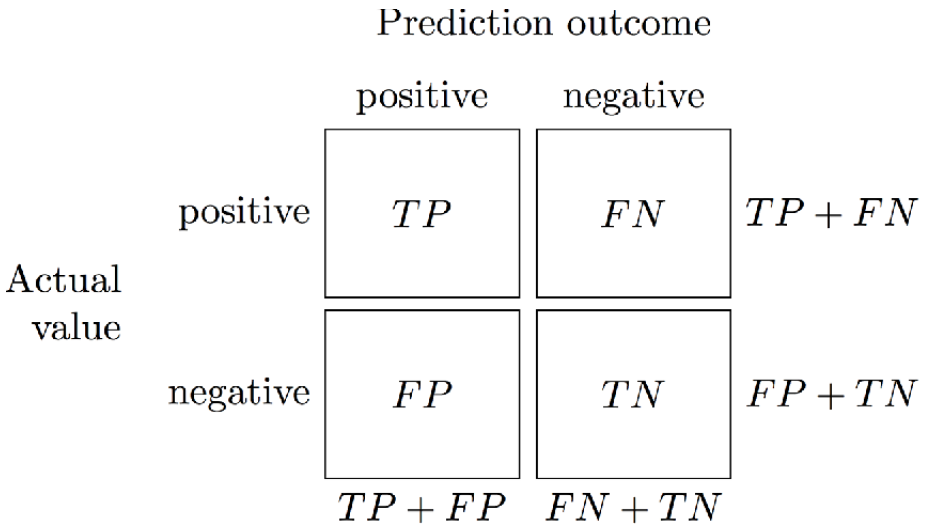


Figure 4.15: Structure of Confusion Matrix.

Area Under Receiver Operating Characteristic Curve

The AUC-ROC curve is a measurement for any classification problem. ROC is the probability curve whereas AUC represents the degree of separability. The area under the roc curve is denoted as the AUC curve. Higher AUC means the model is very good at classifying the predictions correctly. the AUC-ROC curve is plotted with True Positive Rate (TPR) on y-axis, whereas False Postive Rate (FPR) on x-axis.

True Postive Rate:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Specificity:

$$\text{Specificity} = \frac{TN}{TN + FP}$$

False Positive Rate:

$$\begin{aligned} \text{FPR} &= 1 - \text{ Specificity} \\ &= \frac{\text{FP}}{\text{TN} + \text{FP}} \end{aligned}$$

Mean Squared Error (MSE)

Mean Squared Error is the metrics used for regression models. MSE is used as a loss function for least squares regressions. It is calculated by performing the square of the error and then calculating the average of that error. The MSE can be used to get an idea of how close is our fitted line is to a data point.
Mathematical formula for the same is as follow:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Chapter 5

Results

In case of our first dataset (Yahoo S5 Benchmark), the ARIMA model performed quite well as it was able to forecast the values correctly while also filtering the anomalies correctly (Refer to Figure 5.1). The OLS and Ridge Regression was able to predict the anomalies in A2 Benchmark dataset but both of the regression model failed to achieve good score when the model was tested against A1 Benchmark. Using RNN (LSTM Model) we were able to predict the normal values in A2 benchmark and anything which was outside the predicted trend line was considered as anomaly (Refer to Figure 5.3). Overall gaussian distribution for anomaly detection on yahoo dataset was better than any other algorithms.

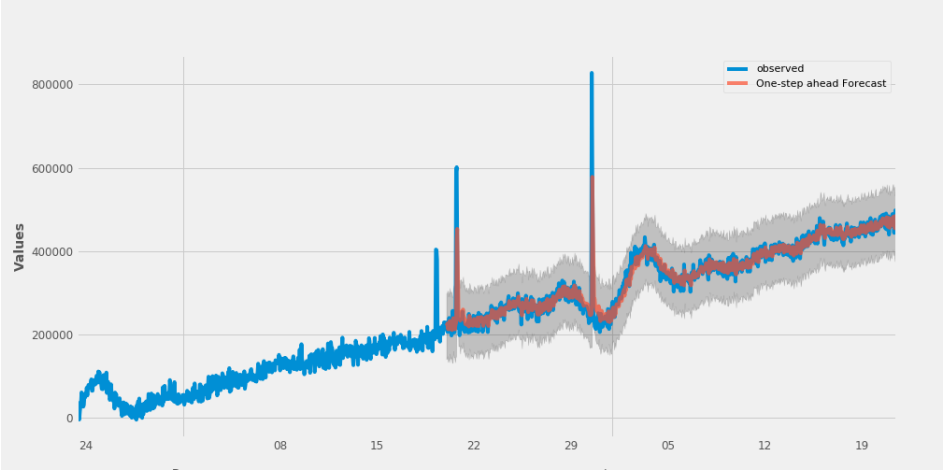


Figure 5.1: ARIMA Forecasting on A2 Benchmark.

	Train	Test	mse	variance	Anomaly present	Anomaly detected	Accuracy score	precision	recall	f1-score
e1	23	18	0.12	-59.05	3	3	1	1	1	1
e2	23	53	0.03	-1.3	17	11	0.9958	1	0.65	0.79
e3	23	24	0.03	-1.38	16	11	0.9965	1	0.69	0.81
e4	23	36	0	-0.28	5	3	0.9986	1	0.6	0.75
e5	53	18	0.03	-15.48	3	2	0.9993	1	0.67	0.8
e6	53	23	0.05	-2.95	19	0	0.9859	0	0	0
e7	53	24	0.01	0.45	16	8	0.9945	1	0.5	0.67
e8	53	36	0.02	-6.19	5	2	0.9979	1	0.4	0.57

Table 5.1: Results on OLS Regression

	Train	Test	mse	variance	Anomaly present	Anomaly detected	Accuracy score	precision	recall	f1-score
e1	23	18	0.06	-28.51	3	2	0.9993	1	0.67	0.8
e2	23	53	0.03	-1.47	17	8	0.9938	1	0.47	0.64
e3	23	24	0.03	-1.57	16	8	0.9945	1	0.5	0.67
e4	23	36	0	-0.24	5	2	0.9979	1	0.4	0.57
e5	53	18	0.03	-13.78	3	2	0.9993	1	0.67	0.8
e6	53	23	0.05	-2.57	19	0	0.9859	0	0	0
e7	53	24	0.01	0.44	16	4	0.9917	1	0.25	0.4
e8	53	36	0.02	-5.41	5	2	0.9979	1	0.4	0.57

Table 5.2: Results on Ridge Regression

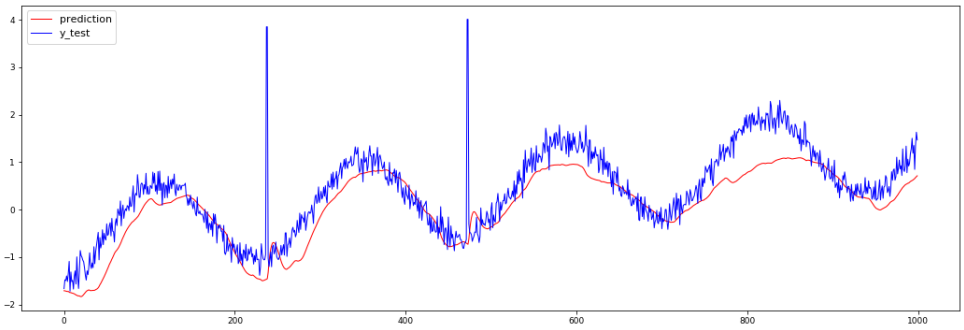


Figure 5.2: LSTM Model (RNN) - Yahoo S5 Dataset..

In case of our next dataset (Numenta Anomaly Benchmark), Cluster based anomaly detection (K-means) performed really well, we divided the time-series into 4 different categories which helped in making the clusters upon that. Combination of categories plus elliptique envelope seemed a good solution for this particular dataset. The evaluation metrics was limited in this case, because of unsupervised model. However, the visualization trends for the same shows that the model was able to correctly identify the anomalies on both Yahoo and Numenta dataset, which means that Cluster based algorithm works well on both Yahoo and Numenta dataset.

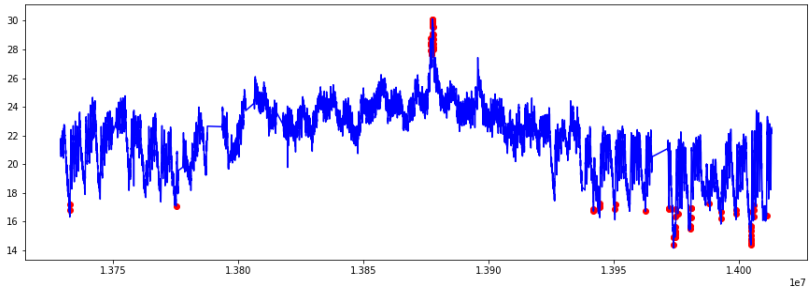


Figure 5.3: Categories + Gaussian Anomaly Detection Algorithm Prediction

For our final dataset (Credit Card Fraud Detection), we performed two different sampling technique to balance the dataset in which Over-Sampling performance was much better than as compared to Under-Sampling. Removing extreme outlier also had a great affect on the model performance. t-SNE results were better than PCA results.

We then used Logistic regression with Over-Sampling (SMOTE) dataset to train the model. We used the ROC Curve for evaluation purpose. The Score for the Logistic Regression model was around 92% and AUC score was 0.8. Refer to Figure 2 and Figure 3

Logistic Regression:

	precision	recall	f1-score	support
0	0.91	1.00	0.95	103
1	1.00	0.88	0.94	86
accuracy			0.95	189
macro avg	0.96	0.94	0.95	189
weighted avg	0.95	0.95	0.95	189

Figure 5.4: Logistic Regression Result on SMOTE Sampling Dataset.

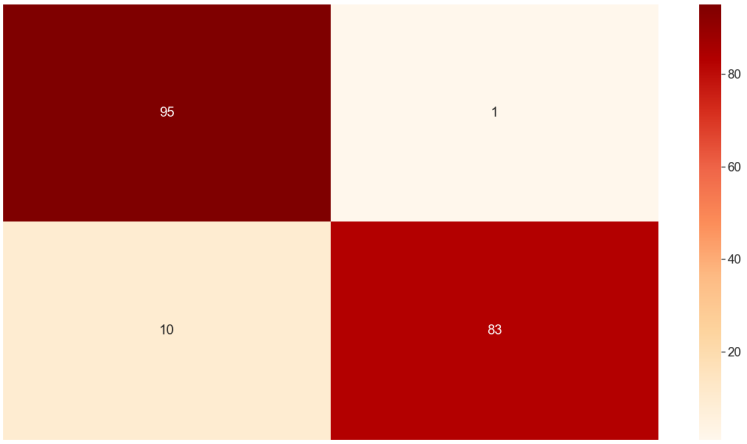


Figure 5.5: Confusion Matrix for Logistic Regression.

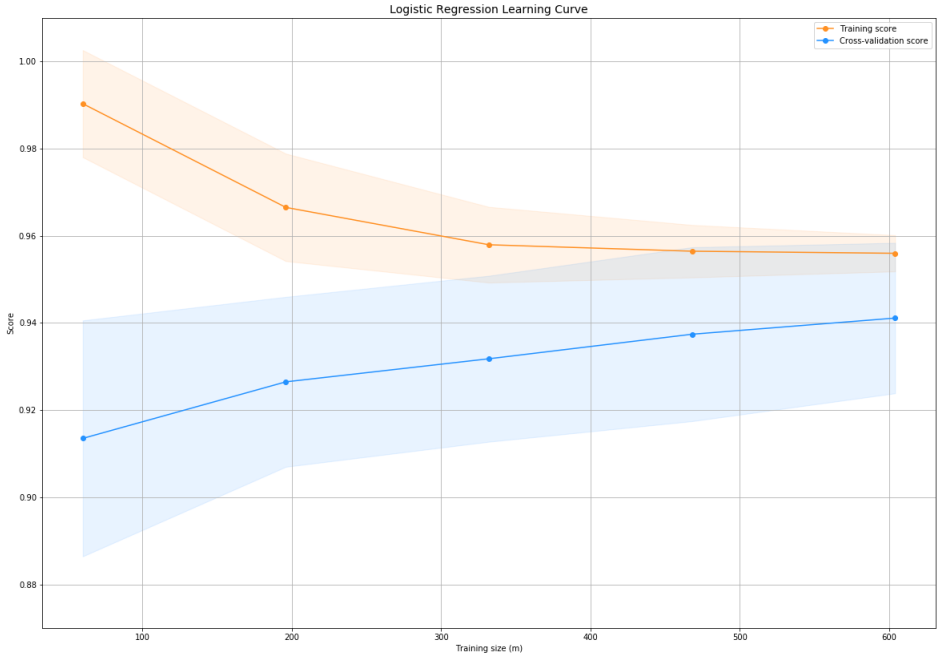


Figure 5.6: Logistic Regression Learning Curve

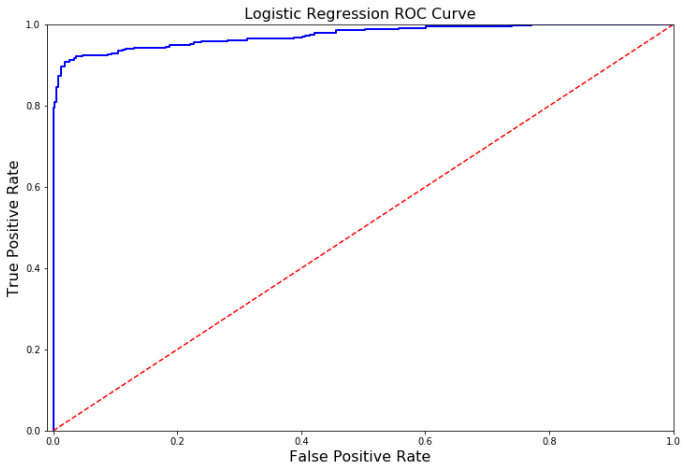


Figure 5.7: ROC Curve for Logistic Regression

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this study, several algorithms were implemented to both pre-process the dataset and to classify any anomalous values. For this research, we used several public dataset such as Yahoo S5 Anomaly Benchmark, Numenta Anomaly Benchmark and Credit Card Fraud Transactions dataset. Each dataset was then thoroughly analysed to understand the nature of the dataset, and pre-processing was done such as generating synthetic values, extracting hourly and weekly categories from the time series dataset, using sampling techniques to balance the dataset, etc. These dataset then were used to train on different algorithms both supervised and unsupervised such as ARIMA, K-means, logistic regression, ridge regression, etc. Each model created was then cross-validated on all the datasets available for benchmarking purpose.

ARIMA model performance was more accurate as compared to other algorithms in case of Yahoo S5 dataset whereas the same model struggled to achieve good score on other two datasets. However, K-means performed well on both yahoo s5 and numenta dataset. It can be said that unsupervised learning technique is more suitable for Yahoo S5 and Numenta dataset. Whereas, in case of credit card dataset, after balancing the dataset using both undersampling and oversampling technique, oversampling method outperformed as it has less number of False positive values as compared to oversampling results. Overall, Logistic regression results was more accurate for our credit card dataset.

6.2 Future Work

All the models implemented above needs to be deployed for real-time anomaly detection over dataset which are not artificial. These type of datasets are quite limited due to several security and privacy factors, which makes the benchmarking of anomaly detection model more difficult. For the last dataset (Credit card Transactions) more algorithms needs to be tested on to check if in case undersampling or oversampling performance is better on it as compared to current best algorithm which is Logistic regression. The undersampling and oversampling was not implemented on yahoo or numenta dataset, in future the same technique can be applied on these two datasets to see if it affects the model performance or if any new algorithms work better on it after the transformation. Data poisoning is something interesting that can be tested on these type of datasets.

Bibliography

[1] A. Pantechovskis, "Determining criteria for choosing anomaly detection algorithm," in *VYTAUTAS MAGNUS UNIVERSITY FACULTY OF INFORMATICS DEPARTMENT OF APPLIED INFORMATICS*, 2019.

[2] . European Central Bank, "Fifth report on card fraud.," Sept. 2018.

[3] A. Lavin, "Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark," in *Numenta, Inc. Redwood City, CA*, 2015.

[4] J. Hawkins, "Hierarchical temporal memory including htm cortical learning algorithms," in *Numenta, Inc. Redwood City, CA*, 2015.

[5] K. Randhawa, C. K. Loo, M. Seera, C. P. Lim, and A. K. Nandi, "Credit card fraud detection using adaboost and majority voting," *IEEE Access*, vol. 6, pp. 14277–14284, 2018.

[6] M. Thill, W. Konen, and T. Bäck, "Online anomaly detection on the webscope s5 dataset: A comparative study," in *2017 Evolving and Adaptive Intelligent Systems (EAIS)*, pp. 1–8, 2017.

[7] O. I. Provotar, Y. M. Linder, and M. M. Veres, "Unsupervised anomaly detection in time series using lstm-based autoencoders," in *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, pp. 513–517, 2019.

[8] D. S. Rosario, "Highly effective logistic regression model for signal (anomaly) detection," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. V–817, 2004.

[9] P. H. Pwint and T. Shwe, "Network traffic anomaly detection based on apache spark," in *2019 International Conference on Advanced Information Technologies (ICAIT)*, pp. 222–226, 2019.

[10] I. Sharafaldin and A. A. Ghorbani, "Eagleeye: A novel visual anomaly detection method," in *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pp. 1–6, 2018.

[11] E. H. M. Pena, M. V. O. de Assis, and M. L. Proença, "Anomaly detection using forecasting methods arima and hwds," in *2013 32nd International Conference of the Chilean Computer Science Society (SCCC)*, pp. 63–66, 2013.

[12] Z. Hasani, "Robust anomaly detection algorithms for real-time big data: Comparison of algorithms," in *2017 6th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–6, 2017.

[13] S. Ahmad and S. Purdy, "Real-time anomaly detection for streaming analytics," 07 2016.

[14] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection: A realistic modeling and a novel learning strategy," *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, pp. 1–14, 09 2017.

[15] A. Dal Pozzolo, O. Caelen, R. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," 12 2015.

[16] B. Lebichot, Y.-A. Le Borgne, L. He, F. Oblé, and G. Bontempi, *Deep-Learning Domain Adaptation Techniques for Credit Cards Fraud Detection*, pp. 78–88. 01 2019.

[17] R. Kumari, Sheetanshu, M. K. Singh, R. Jha, and N. K. Singh, "Anomaly detection in network traffic using k-mean clustering," in *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*, pp. 387–393, 2016.

[18] A. Dal Pozzolo, O. Caelen, Y.-A. Le Borgne, S. Waterschoot, and G. Bontempi, "Learned lessons in credit card fraud detection from a practitioner perspective," *Expert Systems with Applications*, vol. 41, p. 4915–4928, 08 2014.

[19] C. Yin, S. Zhang, J. Wang, and N. N. Xiong, "Anomaly detection based on convolutional recurrent autoencoder for iot time series," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11, 2020.

[20] O. Ibidunmoye, A. Rezaie, and E. Elmroth, "Adaptive anomaly detection in performance metric streams," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 217–231, 2018.

[21] A. Lavin and S. Ahmad, "Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 38–44, 2015.

[22] L. von Werra, L. Tunstall, and S. Hofer, "Unsupervised anomaly detection for seasonal time series," in *2019 6th Swiss Conference on Data Science (SDS)*, pp. 136–137, 2019.

[23] N. Singh and C. Olinsky, "Demystifying numenta anomaly benchmark," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1570–1577, 2017.

[24] C. Huang, G. Min, Y. Wu, Y. Ying, K. Pei, and Z. Xiang, "Time series anomaly detection for trustworthy services in cloud computing systems," *IEEE Transactions on Big Data*, pp. 1–1, 2017.

[25] A. M. Bianco, M. García Ben, E. J. Martínez, and V. J. Yohai, "Outlier detection in regression models with arima errors using robust estimates," *Journal of Forecasting*, vol. 20, no. 8, pp. 565–579, 2001.

[26] V. Chandola, V. Mithal, and V. Kumar, "Comparative evaluation of anomaly detection techniques for sequence data," in *2008 Eighth IEEE International Conference on Data Mining*, pp. 743–748, 2008.

[27] J. Jiang, Y. Yu, Y. Zhu, S. Li, and D. Wan, "Time series outlier detection based on sliding window prediction," *Mathematical Problems in Engineering*, vol. 2014, p. 879736, 2014.

[28] S. A. Shaikh and H. Kitagawa, "Distance-based outlier detection on uncertain data of gaussian distribution," in *Web Technologies and Applications* (Q. Z. Sheng, G. Wang, C. S. Jensen, and G. Xu, eds.), (Berlin, Heidelberg), pp. 109–121, Springer Berlin Heidelberg, 2012.

Appendix

.1 Extra Figures and Tables

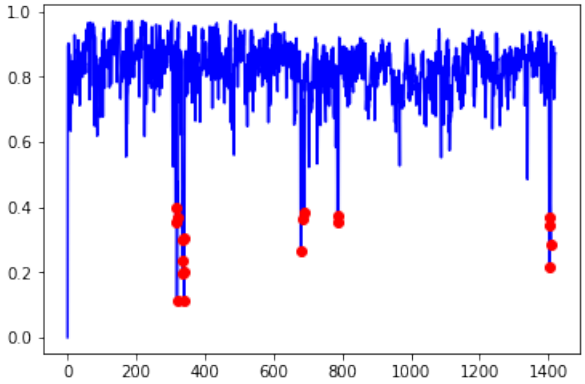


Figure 1: Ridge Regression Prediction on Yahoo S5 Dataset

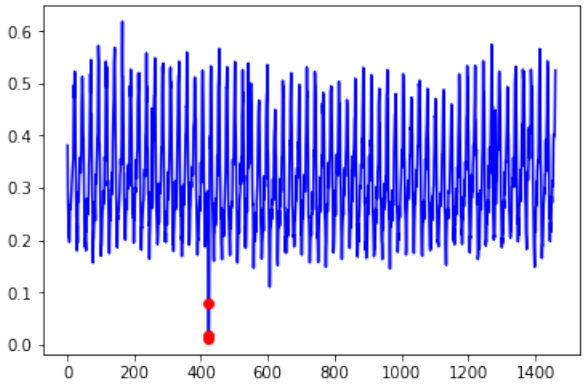


Figure 2: Filtering Anomalies on Yahoo S5 Dataset

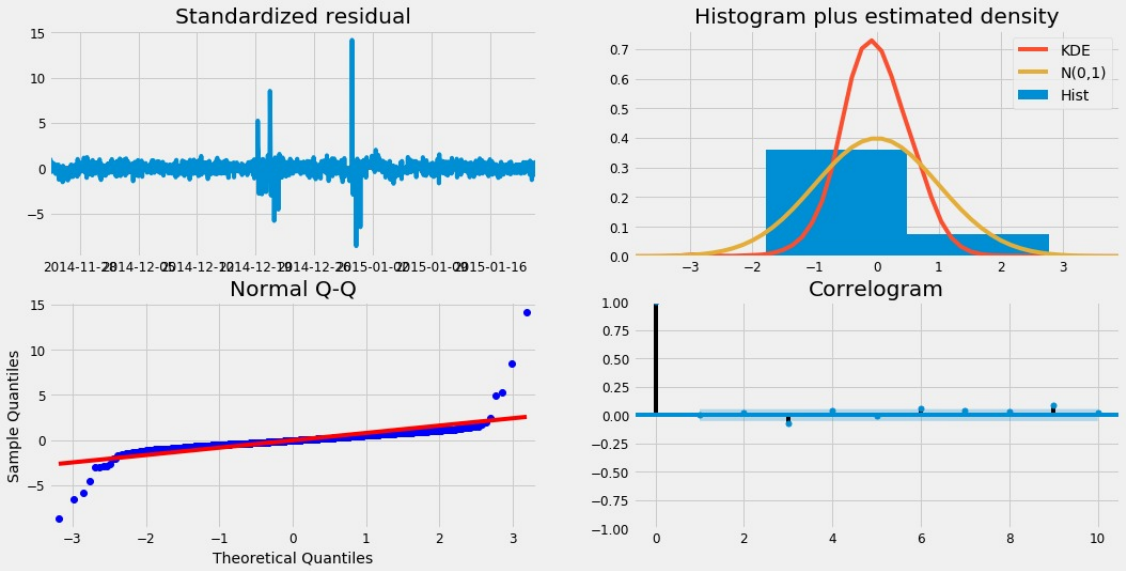


Figure 3: ARIMA Diagnostic

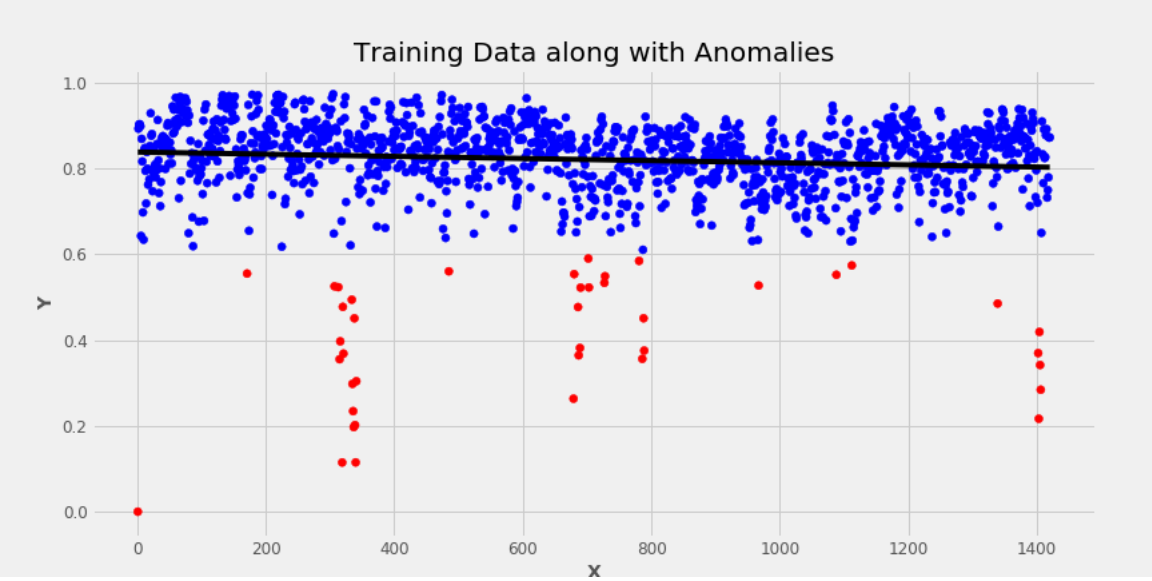


Figure 4: Linear Regression on Yahoo S5 Dataset.