# Analytics of Non-Technical Loss Detection

Thesis for the Degree of Doctor of Philosophy (PhD)

**Khawaja MoyeezUllah Ghori**

**Supervisor: Dr. Laszlo Szathmary**

UNIVERSITY OF DEBRECEN
Doctoral Council of Natural Sciences and Information Technology
Doctoral School of Informatics
Debrecen, 2020

*Hereby I declare that I prepared this thesis within the Doctoral Council of Natural Sciences and Information Technology, Doctoral School of Informatics, University of Debrecen in order to obtain a PhD Degree in Natural Sciences/Informatics at Debrecen University.*

*The results published in the thesis are not reported in any other PhD theses.*

*Hungary, Debrecen, August 2020*

*signature of the candidate*

*Hereby I confirm that Khawaja MoyeezUllah Ghori candidate conducted his studies with my supervision within the Applied Information Technology and its Theoretical Background Doctoral Program of the Doctoral School of Informatics between 2017 and 2020. The independent studies and research work of the candidate significantly contributed to the results published in the thesis.*
*I also declare that the results published in the thesis are not reported in any other theses.*
*I support the acceptance of the thesis.*

*Hungary, Debrecen, August 2020*

*signature of the supervisor*

# Analytics of Non-Technical Loss Detection

Dissertation submitted in partial fulfilment of the requirements for the doctoral (PhD) degree in informatics

Written by **Khawaja MoyeezUllah Ghori** certified computer scientist

Prepared in the framework of the Doctoral School of Informatics
of the University of Debrecen
(Applied Information Technology and its Theoretical Background programme)

Dissertation advisor: Dr. Laszlo Szathmary

The official opponents of the dissertation:

| | | |
|---|---|---|
| | Dr. ................................................ | ............................................. |
| | Dr. ................................................ | ............................................. |
| | Dr. ................................................ | ............................................. |

The evaluation committee:

| | | |
|---|---|---|
| chairperson: | Dr. ................................................ | ............................................. |
| members: | Dr. ................................................ | ............................................. |
| | Dr. ................................................ | ............................................. |
| | Dr. ................................................ | ............................................. |
| | Dr. ................................................ | ............................................. |

The date of the dissertation defence: ……………… 20…

# Abstract

With the ever-growing demand of electric power, it is quite challenging to detect and prevent Non-Technical Loss (NTL) in power industries. NTL is committed by meter bypassing, hooking from the main lines, reversing and tampering the meters. Many countries suffer huge losses in billions of dollars due to NTL in power supply companies. Manual on-site identification of NTL remains an unattractive strategy due to the required manpower and associated cost. The use of machine learning classifiers has been an attractive option for NTL detection.

The literature review has identified the knowledge gap in NTL detection. This gap is surrounded by first finding the best metrics that can identify the top performing classifiers considering the requirements of NTL detection. Secondly, using those metrics finding the best classifiers and the types of the classifiers for NTL detection. Finally, quantifying the impact of feature selection in a real dataset for NTL detection.

Firstly, we compare 14 performance evaluation metrics across the three classifiers and identify the key scientific relationships between them specifically related to NTL detection in a real dataset of an electric supplier containing approximately $80,000$ monthly consumption records. We concluded that recall is the best performance measure for NTL detection. Secondly, we evaluate 15 existing machine learning classifiers for NTL detection across 9 different types. Our work is validated using extensive simulations. Results show that ensemble methods and Artificial Neural Network (ANN) outperform the other types of classifiers for NTL detection. Moreover, we have also derived a procedure to identify the top-14 features out of a total of 71 features, which are contributing in 77% of the prediction in NTL.

In our next contribution, we propose the Incremental Feature Selection (IFS) algorithm, which first uses feature importance to identify the most relevant features for NTL detection and then these features are used to test the three classifiers namely CatBoost, Decision Tree (DT) Classifier and KNN for NTL detection. The results show that using the most relevant features identified by the IFS algorithm, the three classifiers have the same or slightly better efficiency as compared to using all features.

Overall, the thesis has contributed in three main processes of NTL detection, i.e. the feature selection, the identification of suitable machine learning classifiers and their types, and the identification of suitable performance evaluation metrics for NTL detection.

# *Acknowledgements*

In the name of the Almighty Allah, the most gracious the most merciful who gave me courage and strength to finish my project successfully. My prayers are upon his Messenger Prophet Mohammad (peace be upon him) who is our role model in every business of life and who urges to seek knowledge from cot to the grave. My foremost expression of submission and gratitude goes to Allah Almighty without whose grace and mercy; this work would never be possible even in the slightest.

I would like to express my sincere gratitude to my supervisor, Dr. Laszlo SZATHMARY, for giving me the opportunity to commence my PhD studies. It is his trust, continuous guidance and his encouragement and support during difficult times that has enabled me to finish this work. I would like to extend my appreciation to Dr. Marton ISPANY and Prof. Istvan FAZEKAS for their support during my PhD.

No acknowledgment would ever adequately express my obligation to my family for their endless support, love, prayers and good wishes to see me prospering. These are their sacrifices and encouragements for my better education and career which make me what I am today. Special thanks to my father, wife, kids, brothers and sisters.

Finally, I would like to thank my friends Rabeeh Ayaz Abbasi, Muhammad Awais, Muhammad Imran and Ata Ullah for their cooperation to complete my research.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| **Abbreviation** | **Description** |
|---|---|
| NTL | Non-Technical Loss |
| TP | True Positive |
| TN | True Negative |
| FP | False Positive |
| FN | False Negative |
| SVM | Support Vector Machine |
| OPF | Optimum Path Forest |
| DT | Decision Tree |
| MLP | Multi-Layer Perceptron |
| KNN | K-Nearest Neighbors |
| LOF | Local Outlier Factor |
| AUC | Area Under ROC Curve |
| GMM | Gaussian Mixture Model |
| MDS | Multi-Dimensional Scaling |
| BHA | Black Hole Algorithm |
| CNN | Convolutional Neural Network |
| TSR | Three Sigma Rule |
| RMSE | Root Mean Square Error |
| AMI | Advanced Metering Infrastructure |
| FPR | False Positive Rate |
| FNR | False Negative Rate |
| TNR | True Negative Rate |
| TPR | True Positive Rate |
| MCC | Matthews Correlation Coefficient |
| SOM | Self Organizing Maps |

*Dedicated to Prophet Muhammad (Peace be Upon Him) . . .*

# Chapter 1

# Introduction

## 1.1 Introduction

Power companies are considered as one of the important ingredients of a country's financial stability. In today's world, without the supply of electricity, a country can leap back big time in a matter of few days. Therefore, it is not only necessary for the country for a smooth and un-interrupted supply of electricity but it is also obligatory for the company to supply the required amount of electricity in its vicinity. At times, the company can not supply the required electricity. The reason behind this can be interruption of operations or unavailability of the required resources. Another reason is the unexpected amount of electricity required. This is due to the technical and Non-Technical Loss (NTL) in power supply. These problems lead to power load shedding. Line losses are the cause of technical loss. However, NTL happens due to unnatural and criminal (most of the time) attempts to bring down the noted consumed electricity units. Multiple ways are adapted for NTL, like meter by-passing, meter reversing, use of magnet in the meter and false meter reading. The NTL does not only cause a huge monetary loss to the company and the country but it also affects the progress of the country. For example, India loses 4.5 billion USD every year on account of NTL. This loss can range up to 50% of the total electricity produced in developing countries (McDaniel and McLaughlin, 2009). The developed countries, including USA and UK, also suffer a loss of \$1 – \$6 billion annually (Alam et al., 2004). Similarly, Brazil suffers 4.5 billion dollars annually due to NTL (Bhat et al., 2016). Pakistan's economy is also suffering from 0.89 billion dollars annually on account of NTL (Hussain et al., 2016).

Over the last decade, the research community has been actively participating in an attempt to bring down the occurrences of NTL in power industry. For this, the use of physical devices as well as the data analytics on consumption patterns are major sources of NTL detection schemes. Many developed countries have installed Advanced Metering Infrastructure (AMI) in their network. The AMI is a smart metering system with a two-way communication between the supplier and the consumer that helps in better management of consumption history. The use of AMIs enables the company to record the consumption history over a different intervals of time like hourly, twelve-hourly, daily and bi-monthly consumption recordings. However, many under-developed countries still use the traditional meters for which monthly manual meter reading is required. Due to the use of old metering structure, the chances of NTL grow as the use of meter by-passing, magnet and false meter reading is easy in old metering infrastructure. In order to detect NTL, companies have used separate physical observer devices in meters and distribution poles which calculate the difference of energy supplied and used. However, this approach incurs heavy monetary cost. Another widely adapted method for NTL detection is the use of machine learning classifiers on a dataset of hourly, daily or monthly consumption records. These classifiers help in identifying potential theft which otherwise is a costly manual effort. However, still there is a need to improve the way we use classifiers and their evaluation metrics for NTL detection. This knowledge gap is highlighted in the next section.

## 1.2 Knowledge Gap in Existing Techniques for NTL Detection

A substantial literature is available on the techniques used for NTL detection. However, there are open challenges and knowledge gaps which are still needed to be addressed. These challenges are discussed in the following subsection and represented in Figure 1.1.

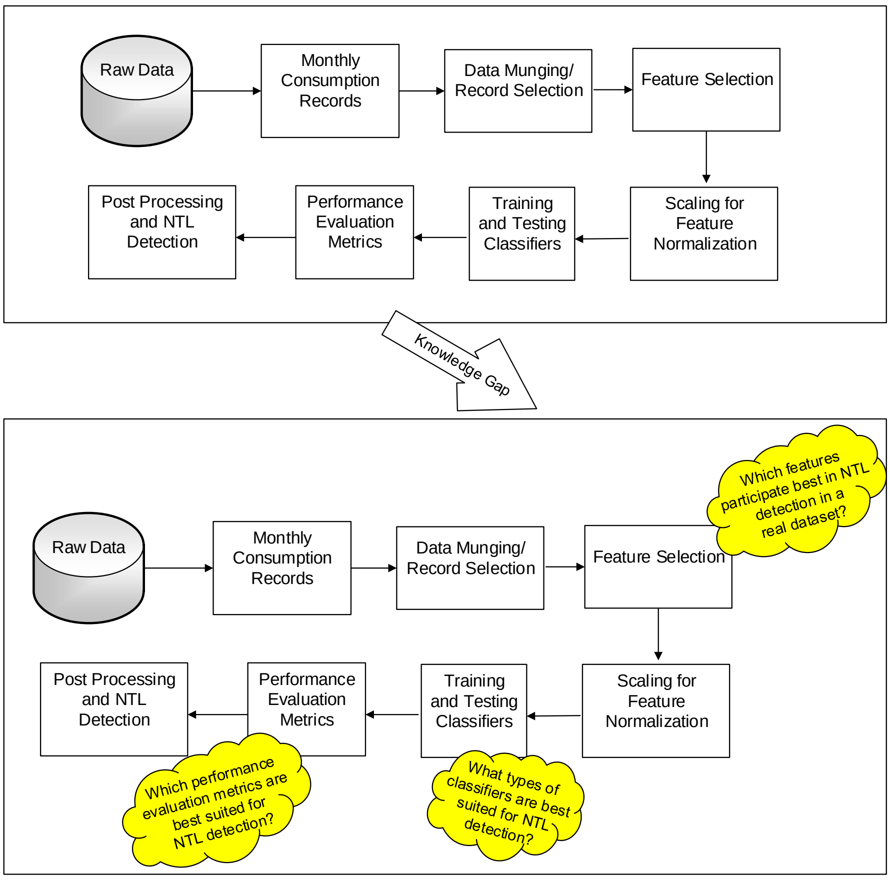
Raw Data
Monthly Consumption Records
Data Munging/ Record Selection
Feature Selection
Scaling for Feature Normalization
Training and Testing Classifiers
Performance Evaluation Metrics
Post Processing and NTL Detection
Knowledge Gap
Raw Data
Monthly Consumption Records
Data Munging/ Record Selection
Feature Selection
Which features participate best in NTL detection in a real dataset?
Scaling for Feature Normalization
Training and Testing Classifiers
Performance Evaluation Metrics
Post Processing and NTL Detection
Which performance evaluation metrics are best suited for NTL detection?
What types of classifiers are best suited for NTL detection?

FIGURE 1.1: Knowledge Gap in Existing Techniques for NTL Detection

### 1.2.1 Identification of the Best Performance Evaluation Metrics for NTL Detection

In a real dataset, the ratio of records with theft and normal consumption is vastly imbalanced. That is, the number of records of normal consumption is very high whereas the number of records of theft instances are very small. The problem becomes interesting when the focus is on least representative records, i.e the theft records. This type of problem belongs to class imbalance problem where the ratio of representation of the two classes is highly imbalanced. For this reason, not every metric is useful in evaluating the performance of the classifiers used. Keeping in mind the number of normal consumption identified as theft and the number of theft records identified as normal, there is a need to prioritize the available metrics which can best describe the performance of the classifiers used in NTL detection.

### 1.2.2 Comparison of Performance of Different Types of Classifiers for NTL Detection

A range of machine learning classifiers is available for training and testing. These classifiers belong to different types. For e.g, Gaussian Naive Bayes (John and Langley, 1995) and Bernoulli Naive Bayes (McCallum, Nigam, et al., 1998) classifiers belong to the type of Naive Bayes Classifiers. Similarly, Random forest (Breiman, 2001), AdaBoost (Freund and Schapire, 1997) and CatBoost (Prokhorenkova et al., 2018) belong to the type of ensemble methods. Efforts have been made to compare different classifiers for NTL detection but there is a need to compare not only the performance of the individual classifiers but also the comparison of the types of the classifiers for NTL detection in some real dataset. Also, a recently developed CatBoost, LGBoost and XGBoost classifiers are also needed to be tested for NTL detection in a real dataset.

### 1.2.3 Identification of Best Features for NTL Detection in a Real Dataset

Feature selection is an important process before the selected data is used for the training and testing of the classifiers. Multiple techniques for feature selection are available. For NTL detection, there is a need to identify most relevant features in a real dataset. In other words, the feature importance of every feature in participating in NTL detection is needed to be precisely monitored and evaluated for some real dataset. The performance of using only selected features is also needed to be compared with the performance when all features are used.

## 1.3 Aims of the Thesis

The main aim of the thesis is to first find the best metrics that can evaluate classifiers considering the requirements of NTL detection. Then, use those metrics for better evaluation of the classifiers and the types of the classifiers. The objectives of the thesis are outlined below:

1. Work on a real dataset for the identification of best metrics which can evaluate different machine learning classifiers considering the requirements of NTL detection.

2. Compare the performance of the different machine learning classifiers used for NTL detection.

3. Compare the performance of the types of machine learning classifiers for NTL detection.

4. Compare the performance of the recently developed CatBoost, LGBoost and XGBoost classifiers for NTL detection in a real dataset.

5. Identify the most relevant features for NTL detection in a real dataset.

## 1.4 Thesis Overview

The thesis is comprised of six chapters. The details of the chapters are discussed in the following subsections and the overview is presented in Figure 1.2.

### 1.4.1 Chapter 1

Many countries face huge financial losses on account of NTL in power industry. Chapter 1 first introduces the problem of NTL detection and its demand in today's world. Then, it highlights the knowledge gap in recent practices of NTL detection. It further discusses the contributions of this dissertation to fill the knowledge gap in the field of NTL detection. Lastly, it presents the aims and the outline of the thesis.

### 1.4.2 Chapter 2

A comprehensive literature review is always helpful in finding the latest trends and research contributions in a specific domain. Chapter 2 presents the state of the art methodologies used in NTL detection. It presents a comprehensive taxonomy of the techniques used and a detailed comparative study of those techniques. The techniques involve data-oriented consumption and additional data profiling, network oriented techniques and hybrid of data and network based techniques. It also highlights the limitations of the current on-going efforts in NTL detection. Lastly, this chapter discusses the theoretical description of all the classifiers and the performance evaluation metrics used in this thesis.

### 1.4.3 Chapter 3

In order to contribute in the problem of NTL detection in power industry, a real-world dataset is necessary on which pre-processing steps along with the training and testing of multiple machine learning techniques can be applied. Chapter 3 presents a detailed description of the real dataset and the necessary pre-processing steps that are required in the proposed methodology. It also contains the

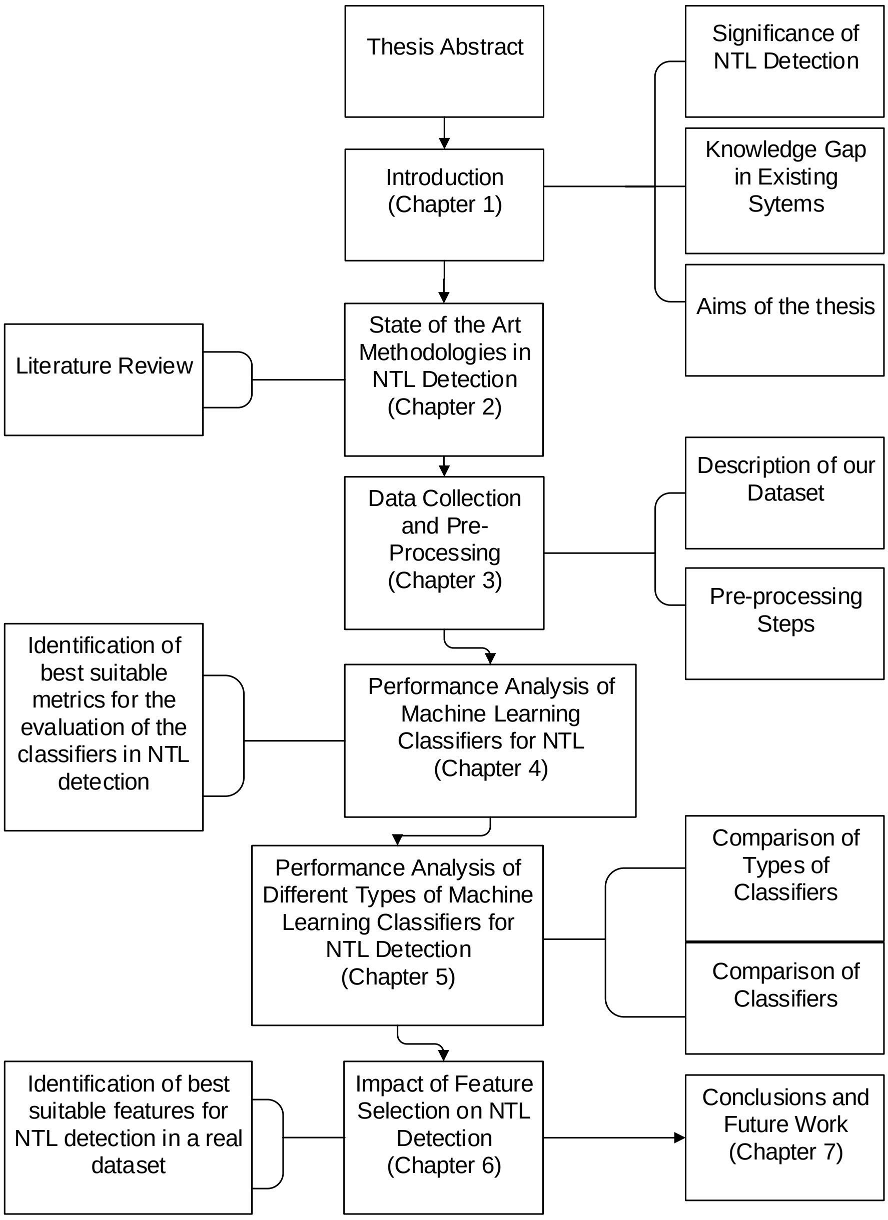
Thesis Abstract
Introduction (Chapter 1)
Significance of NTL Detection
Knowledge Gap in Existing Sytems
Aims of the thesis
Literature Review
State of the Art Methodologies in NTL Detection (Chapter 2)
Data Collection and Pre-Processing (Chapter 3)
Description of our Dataset
Pre-processing Steps
Identification of best suitable metrics for the evaluation of the classifiers in NTL detection
Performance Analysis of Machine Learning Classifiers for NTL (Chapter 4)
Performance Analysis of Different Types of Machine Learning Classifiers for NTL Detection (Chapter 5)
Comparison of Types of Classifiers
Comparison of Classifiers
Identification of best suitable features for NTL detection in a real dataset
Impact of Feature Selection on NTL Detection (Chapter 6)
Conclusions and Future Work (Chapter 7)

FIGURE 1.2: Thesis Overview

features, their description and the transformation needed before the dataset is used for training and testing the classifiers.

### 1.4.4 Chapter 4

As the problem of NTL detection belongs to the class imbalance domain, a bias is observed between the number of observations for the normal consumption and the number of observations for the theft cases. Therefore, considering the requirements of NTL detection, there is a need to identify the metrics that are best suited for the performance evaluation of the classifiers. Chapter 4 compares the performance of three classifiers using 15 performance evaluation metrics. Finally, it presents a detailed analysis of the pros and cons of each metric for NTL detection. Thus, it identifies the relationship that exists between different metrics considering the specific requirements of the NTL detection. This contribution was published in (Ghori et al., 2020b).

### 1.4.5 Chapter 5

In recent years, different machine learning classifiers have been tested in NTL detection. However, there is a need to identify the types of the classifiers that perform best in NTL detection. Chapter 5 presents this contribution in comparing not only the performance of the classifiers but also the performance of the types of the classifiers. It also introduces the recently developed CatBoost, LGBoost and XGBoost classifiers for NTL detection in the real dataset. Lastly, with the help of extensive simulations, it highlights the best classifiers and the best type of classifiers for NTL detection in the real dataset. In addition, this chapter compares the performance of deep learning with the performance of the other classifiers. This contribution was published in (Ghori et al., 2019).

### 1.4.6 Chapter 6

Not only the identification of the relevant records is important for correctly identifying the potential theft but the identification of the relevant features is equally important. This is because not all the features are relevant to the identification of NTL. Chapter 6 highlights this contribution of the identification of the best features for NTL detection in a real dataset. It also introduces our proposed Incremental Feature Selection (IFS) algorithm for feature selection. Finally, it compares the success ratio of NTL detection using all features with the success ratio of NTL detection using selected features from the IFS algorithm. This contribution was published in (Ghori et al., 2020a).

### 1.4.7 Chapter 7

Chapter 7 concludes with the summary of the contributions of this work and discusses the future directions in the field of NTL detection.

## 1.5 Thesis Points

The thesis points are summarized below:

1. The thesis contains a comprehensive literature review which has enabled to identify the limitations of recent works in NTL detection. (See Chapter 2).

2. A total of 14 performance evaluation metrics are analyzed for NTL detection using different classifiers. We found that recall should be given higher priority for NTL detection, and random forest is the better algorithm for it having the highest recall. (See Chapter 4).

3. We have performed the testing of 15 machine learning classifiers belonging to 9 different types. The MLP classifier was found the best individual classifier with respect to recall, and ANN was found the best type of the classifiers for NTL detection. (See Chapter 5).

4. We proposed a novel framework to identify relevant features for NTL detection by using the Incremental Feature Selection (IFS) algorithm, which identifies the most relevant features for NTL detection in a real dataset using feature importance. The

results have shown that with the use of the IFS algorithm, recall and F-Measure of KNN is increased by 120% and 60%, respectively, while the training time of KNN is reduced by 90%. (See Chapter 6).

# Chapter 2

# State of the Art Methodologies in NTL Detection

## 2.1 Literature Review

Big data analytics is frequently used in diverse domains of everyday life. It strives to solve realistic problems by applying machine learning algorithms and data mining approaches. The applications include fraud detection (Jain and Gupta, 2019), problem handling of unstructured data (Amalina et al., 2019), disease comorbidity prediction (Lakshmi and Vadivu, 2019), Internet of Things (IoT) (Rehman et al., 2019), Industrial Internet of Things (IIoT) (Rehman et al., 2018), real-time anomaly detection (Ariyaluran Habeeb et al., 2019; "Real-time big data processing for anomaly detection: A Survey" 2019), preventive medicine using big data (Razzak, Imran, and Xu, 2019), and event detection (Saeed et al., 2019).

During the last few years, the research community has paid attention to the problem of NTL detection. To encounter this problem, supervised, unsupervised and semi-supervised learning methods have been used. Some of the authors have used customers' consumption history while others have used the grid and network data. Effort has also been made to use both types of data, i.e. consumers' consumption profile as well as the grid data which may contain current and voltage information supplied to different areas. At times, some additional data are also merged to the consumption data to see the effect of hit ratio of NTL detection. This additional data may comprise of environmental and temperature readings.

NTL identification is an application of fraud detection (Han and Xiao, 2019). A survey of the existing techniques to detect NTL can be found in (Papadimitriou et al., 2017). The study has categorized the techniques handling NTL into data-oriented, network oriented and hybrid techniques. The data-oriented techniques use consumers' consumption patterns to predict NTL. These techniques can further be divided into supervised, semi-supervised and unsupervised learning paradigms. Supervised learning methods are used when the class label of fraud and non-fraud is provided. Example of supervised learning is SVM. Semi-supervised learning methods are used when only one class label is known and the other label is not definite. Example of semi-supervised learning is anomaly detection. Unsupervised learning methods are used when the class labels are not used at all. Clustering algorithms are the examples of unsupervised learning. Network-oriented techniques include usage of the network data and smart meters, which are used to check electric balance with respect to the grid. The authors of (Papadimitriou et al., 2017) have stated that network-oriented techniques are good at detecting NTL in a specific area but fail to identify specific fraudulent consumers. Hybrid techniques use advantages of both techniques where network data is used to locate the fraudulent area and consumption data is used to identify fraudulent consumers. They have listed TP, TN, FP, FN, recall, FPR, recognition rate and Bayesian detection rate as the main performance evaluation metrics. Alongside, they have discussed the roles and responsibilities of the concerned authorities to tackle NTL.

Another comprehensive survey for the challenges of NTL detection can be found in (Glauner et al., 2016b). The authors have compared multiple techniques which are applied in NTL detection. These include expert system, machine learning, SVM, Neural network, fuzzy logic, genetic algorithm, optimum path forest, and rough sets. They have also compared different search techniques for feature selection of customers' master data. The paper identifies some challenges which are still needed to be thoroughly dealt with. For example, the identification of a correct percentage of under sampling of majority class, a need of a thorough comparative study for different techniques dealing imbalance domain, a need of a metric to compare regression with classification problems and creation of a benchmark dataset.

Another survey on NTL detection can be found in (Messinis and Hatziargyriou, 2018). The authors of this paper have also categorized the techniques used in NTL into Data Oriented, Network Oriented and Hybrid Techniques. They have categorized the consumption data into multiple categories like time series data, raw data and geographical data. The authors have also discussed the classification and clustering techniques used for NTL and the performance evaluation metrics used to evaluate them.

On the basis of the type of data used to detect NTL, the strategies can be categorized into three main types namely Data-Based Techniques, Hybrid Techniques and Network-Based Techniques. The Data-Based Techniques are further divided into two sub-types namely Additional Data Profile and Consumption Profile while the Network-Based Techniques are divided into three sub-types namely state examination, current flow examination and sensor installation. Hybrid is a combination of the two techniques, that is, Data-Based Techniques and Network-Based Techniques. Consumption profile involves detecting NTL using only the consumption data collected from the meters installed at the consumer end. The consumption data can take monthly, daily, hourly or half hourly readings. Additional data based techniques not only use consumption data but they also use data from outside of the system, for example, climate and temperature data. Network based techniques involve detecting NTL using the difference between the total electricity supplied and the total electricity billed. These techniques also use the grid data. Efforts have been made in every category to detect NTL. A complete categorical division of the types of strategies and algorithms used in NTL is shown in Figure 2.1.

### 2.1.1 Data-Based Techniques

Two different types of data profiles are used for NTL detection. Consumption profile contains half hourly, hourly, daily or monthly consumption records of consumers which is used to detect a potential

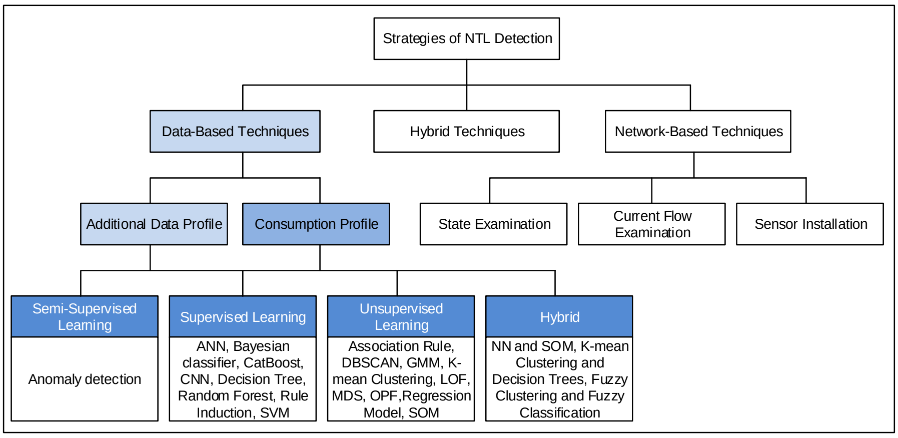
Strategies of NTL Detection
Data-Based Techniques
Hybrid Techniques
Network-Based Techniques
Additional Data Profile
Consumption Profile
State Examination
Current Flow Examination
Sensor Installation
Semi-Supervised Learning
Anomaly detection
Supervised Learning
ANN, Bayesian classifier, CatBoost, CNN, Decision Tree, Random Forest, Rule Induction, SVM
Unsupervised Learning
Association Rule, DBSCAN, GMM, K-mean Clustering, LOF, MDS, OPF,Regression Model, SOM
Hybrid
NN and SOM, K-mean Clustering and Decision Trees, Fuzzy Clustering and Fuzzy Classification

FIGURE 2.1: Strategies of NTL detection

NTL. Additional data profile is also merged with consumption profile in an effort to increase accuracy in predicting NTL. The additional data profile may contain environmental and temperature features.

#### 2.1.1.1 Consumption Profile

To address the problem of NTL, the research community has used unsupervised, supervised, semi-supervised and even a combination of supervised, unsupervised and semi-supervised learning methods. Research contributions of some of them are discussed below.

## Unsupervised Learning

The authors of (León et al., 2011) have used consumption data collected from Endesa Distribucion, a power supply company in Spain. They have used association rule mining to cluster a group of customers responsible for electricity theft. The use of association rule mining has enabled them to perform an on-site inspection of a filtered few hundred consumers out of thousands of consumers. They have used support, confidence, TP, TN, FP and FN metrics to evaluate their result. They have claimed a 7% to 20% increase in detecting NTL. However, the paper has not discussed about the strategy used for feature selection process. In (Singh and Yassine, 2018), the authors have also used association rule mining by proposing an algorithm of their own which generates frequent patterns of the use of appliances. They have claimed to find associations between appliances of home and time series. Their work can further lead to NTL detection by filtering out those instances which disobey frequent patterns of a specific household area. They have stated that their results outperformed SVM and Multi-layer perceptron (MLP).

Benford curve, hierarchical clustering and Multi-dimensional scaling (MDS) are used in (Sánchez-Zuleta, Fernández-Gutiérrez, and Piedrahita-Escobar, 2017) to study the characteristics of consumption for a better detection of NTL in two companies. One of their findings is that in company 1, fraudsters have a different curve as compared to normal consumers with respect to Benford curve. No such indication is observed in company 2. This behavior is also indicated by the fact that using decision trees, company 1 has a good classification for fraudsters as compared to company 2.

Sharma et al. (Sharma et al., 2017) have used the concept of local outlier factor (LOF) in density-based spatial clustering of applications with noise (DBSCAN) clustering algorithm to identify unusual load patterns in two datasets from USA and India. LOF is the ratio of density of a data point to the density of its k-nearest neighbors. A higher value of LOF shows that there is a noticeable difference between the densities of the point and its neighbors reflecting the point to be suspicious. They have used Silhouette coefficient and Davies Bouldin index to evaluate their technique but did not compare them with other clustering algorithms. A similar approach is used in (Zheng et al., 2017). The authors have proposed a distance matrix to observe the unusual profiles of consumers. They have used Area Under ROC Curve (AUC), accuracy and F1 measures to evaluate their proposed model and compared it with Gaussian mixture model (GMM), k-means and DBSCAN. Their results show that their technique outperformed the already known techniques. A signal processing technique is used in (Avila, Figueroa, and Chu, 2018) for feature extraction and selection in the same dataset. In addition, a smoothing spline function is used for outlier detection in consumers data. The paper has also proposed an undersampling boosting algorithm for NTL detection. Using seven performance evaluation metrics, the paper has argued that the proposed framework has performed better than the available NTL detection techniques. However, the paper does not discuss the features representing the real data. In (Sharma and Singh, 2015), the authors have also proposed a density based clustering algorithm called DBMSCAN. It identifies low and peak loads which in turn help in detection of irregular consumption. The algorithm encounters anomalies by introducing irregularity variance. The authors have used silhouette coefficient for the comparison of DBMSCAN with the traditional DBSCAN and stated that their algorithm has outperformed DBSCAN.

Another unsupervised method, Optimum-path forest (OPF), is used in (Júnior et al., 2016) to detect NTL in a Brazilian electricity dataset. The authors have also used semi-supervised learning method of anomaly detection that has the information of only one class. They have compared the accuracy of both techniques with SVM, GMM, *k*-means, one-class SVM, Birch and affinity propagation. Accuracy, F-Measure and standard deviation are used as efficiency measure for the classifiers. The paper stated that OPF and anomaly detection techniques (i.e. semi-supervised learning) outperformed others. The paper however does not discuss any feature selection technique. Yeckle et al. (Yeckle and Tang, 2018) have used seven different outlier detection techniques to identify the occurrence of NTL in an Irish dataset. They have also performed k-means clustering algorithm in the pre-processing step to cut off the number of transactions per day to three. They have tested the performance of the outlier detection techniques using AUC and claimed that reducing the number of meter readings by using k-means clustering has helped improving the performance of AUC.

It is important to identify the best combination of features that can participate in predicting NTL. For this, different feature selection techniques have to be deployed to the dataset before it is used for NTL detection. In (Ramos et al., 2016), the authors have used the Black Hole Algorithm (BHA) for feature selection in two different datasets taken from the Brazilian electric supply company. The datasets contain commercial and industrial consumption records. They have used mean accuracy rates for each class to compare the results of NTL detection with feature selection and without feature selection. The paper concludes that with feature selection, the hit ratio increased by 20% for industrial records and 6% for commercial records. However, the paper does not discuss feature relevance of each feature with respect to the target variable. In order to find the most relevant features for NTL detection, a strategy is needed that can systematically add features and evaluate their performance in the feature selection process. Ramos et al. (Ramos et al., 2011b) have used a combination of harmony search algorithm and Optimum Path Forest (OPF) for feature selection in a dataset of an electric supplier from Brazil. They have used commercial and industrial datasets with eight features which were identified after the feature selection process. The paper states that their approach has not only reduced the overall execution time but also has identified useful features. Accuracy is the only metric used for the evaluation despite the data being imbalanced (a situation where accuracy is not the right measure for evaluation). In their next contribution (Ramos et al., 2011a), they have compared three evolutionary techniques for feature selection namely harmony search, particle swarm optimization (PSO) and gravitational search. The dataset used is the same as in (Ramos et al., 2011b). The authors have concluded that using a combination of PSO-OPF, the feature recognition rate can be improved from 92% to 98%. In the latter contribution, the authors have not described the details of classifiers and their performance evaluation for NTL detection.

## Supervised Learning

Similar attention is made in dealing NTL detection through supervised learning techniques. For example, Zheng et al. (Zheng et al., 2018) have experimented wide and deep convolutional neural network (CNN) in a dataset collected from a Chinese electricity company. Wide framework of the neural networks handles the 1-D consumption records of each consumer, while the deep framework maintains weekly consumption. They have used AUC as the evaluation metric to compare their work with existing classifiers like SVM, logistic regression, random forest, and three sigma rule (TSR). They observed that their model outperforms these classifiers. In (Ford, Siraj, and Eberle, 2014), artificial neural networks have been used to predict electric theft detection in a relatively smaller dataset collected from the Irish Social Science Data Archive Center. The authors have trained the neural network on three different situations and consequently have stated three observations. One of the observations is that consumers' consumption behavior can be predicted a year ahead. The other observation is that the training of a neural network on the data of three consecutive weeks can predict consumer's consumption behavior for the fourth week. Their final observation is that the consumption patterns can also be predicted in the same weather season. They have used TP, TN, FP and FN to measure the performance of the neural network. The authors extended their work in (Cody, Ford, and Siraj, 2015) to train and test the Irish dataset using M5P decision tree on the same situations. They have used root mean square error (RMSE) to measure the closeness of predicted and actual values. RMSE values are found within the threshold for all three situations.

The authors of (Coma-Puig et al., 2016) have used a dataset of a company providing electricity and gas in Spain. They have used a combination of different machine learning techniques to predict NTL in electricity and theft attempts in gas sector. This includes Naive Bayes, KNN, decision trees, neural networks, SVM, random forest and AdaBoost. Their framework has a feature which auto-updates the results of on-field inspection in the database resulting the framework to be adaptive to new theft patterns over a period of time. They have used precision as the only performance evaluation metric.

During the last few years, advancements in deep learning have opened a lot of application areas (Hayat et al., 2019). One of its application areas which still needs attention of research community is NTL detection. The authors of (Bhat et al., 2016) have tested convolutional neural network, auto encoders and long short-term memory networks for NTL detection in a relatively smaller dataset containing occurrences of NTL. Experimental results demonstrate that deep learning-based strategies have outperformed decision trees, random forest, and neural networks in terms of various performance metrics such as precision, recall, F1 and receiver operating characteristic (ROC) curve.

In (Chatterjee et al., 2017), the authors have used deep recurrent neural networks to identify NTL. The data used is related to advanced metering infrastructure (AMI). It is taken from Australian Governments Department of Industry, Innovation and Science. As AMI's data is sequential with respect to time, so recurrent neural network is applied to it. The metric used to evaluate the performance is accuracy which is measured to be 65.3% for a neighborhood. However, it does not use any other performance evaluation metric which may help for a better understanding of NTL detection.

Fuzzy logic is used in (Spirić, Stanković, and Dočić, 2018) to detect potential electricity theft consumption. The authors have used consumption data from 2003 to 2017 in a series of five decades incrementing a year every time starting from 2003. Fuzzy suspicions are created based on the relationship of consumption between time-series data. Fuzzy logic is then used to calculate suspicion value for each consumer. If it passes a certain threshold, the consumer is considered a suspicious consumer. This work has shown a 14% of success percentage in finding the theft cases but it claims that the percentage of success will be increased after on-site inspection. A similar type of work is presented in (Viegas, Esteves, and Vieira, 2018) where authors have used fuzzy-based distance to check whether a consumer′s distance has significantly crossed a consumption prototype. They have used consumption records of four thousand Irish households. The authors have claimed a true positive rate of 63.6% and a false positive rate of 24.3%. Glauner et al. (Glauner et al., 2016a) have used Boolean rules, fuzzy logic and SVM to detect NTL in a dataset of around a million customers while analyzing their monthly consumption patterns. The results show that optimized fuzzy logic and SVM outperformed Boolean rules. They have compared the classifiers with performance evaluation metrics like true positive, true negative, false positive, false negative, recall and specificity. However, the relationships that may exist between these metrics regarding NTL detection are not sufficiently discussed.

Recently developed ensemble methods namely CatBoost, LGBoost and XGBoost are tested in (Punmiya and Choe, 2019). The authors have used an Irish dataset that contains half-hourly meter readings for 420 days. They have generated six theft cases in the dataset to balance out the minority class. They have concluded that LGBoost and CatBoost outperformed XGBoost with respect to detection ratio, while LGBoost performed better than CatBoost and XGBoost with respect to False Positive Rate (FPR). However, they have not compared the three classifiers with other known classifiers. Moreover, their dataset is synthesized with equal positive and negative class representations. This, of course, does not reflect the real-world scenario where observations of positive class are very small as compared to the negative class. There is a need that these classifiers should also be tested on a real-world dataset. A set of classifiers have been used as ensembles to detect frauds in an electricity supply company in Uruguay by Di Martino et al. (Di Martino et al., 2012). They have claimed that a one-class SVM, CS-SVM, Optimum Path Forest (OPF) and C4.5 combined as ensembles have given good measures as compared to applying them individually. The classifiers are compared using accuracy, recall, precision and $F$ value. However, this paper does not discuss the impact of using these metrics for NTL detection.

A more widely used set of techniques for NTL detection belong to the traditional data mining and machine learning classifiers. For example, a dataset of an electric supplier from Honduras is used in two research contributions. Figueroa et al. (Figueroa et al., 2017) have used random oversampling and undersampling in the preprocessing step to handle the imbalance behavior of the target variable. They have used three classifiers namely linear SVM, non-linear SVM and a multi-layer perceptron neural network for NTL detection in a dataset collected from an electric company operating in Honduras. They have used under-sampling and over-sampling strategies to handle the imbalance ratio of fraud and non-fraud instances. Additionally, eight performance evaluation metrics are used to compare performances of the classifiers. The metrics include accuracy, recall, precision, specificity, AUC, $F_\beta$, $F_1$ and Matthews Correlation Coefficient (MCC). However, the paper does not discuss the relationships between these metrics specifically for NTL detection. These relationships may be used to find appropriate combination of metrics for NTL detection. A total of 49 features are used to test the classifiers but the paper does not discuss any feature selection strategy deployed. The paper concluded that the oversampling strategy produced better results. The authors of (Nagi et al., 2010) have used

SVM to identify NTL in a dataset that is having a highly uneven distribution of class labels. They have claimed a hit rate increase from 3% to 60%. This work is focused on identification of NTL where abrupt changes of users' consumption patterns are found but does not discuss situations where changes are observed gradually. It compares the results of NTL using accuracy and hit rate but does not discuss any relation between them.

### Hybrid of Supervised and Unsupervised Learning

Some of the authors have applied supervised and unsupervised learning as a sequence of operations to deal with NTL. For example, in (Guerrero et al., 2018), the authors have taken a dataset of a Spanish electricity supply company and performed two modules. In the first module, they have used artificial neural networks to filter the consumers in a pre-processing step. In the next module, they have tested classification and regression tree (supervised learning) along with Self Organizing Maps (SOM), which is a technique used in unsupervised learning. The authors have claimed a three times rise of accuracy as compared with manual inspection. Similarly, authors of (Peng et al., 2016) have performed a mixture of unsupervised and supervised learning techniques in a dataset collected from a Chinese electric company. In the first step, they have performed k-means clustering algorithm to form different clusters of consumers based on their consumption patterns. In the second step, they have performed a reclassification step by applying decision tree, random forest, SVM and KNN to the filtered consumers obtained from the first step. The paper concludes that the classification step overcomes the weaknesses that appeared in the clustering step.

A score-based approach is used in (Terciyanli et al., 2017), which combines fuzzy clustering and fuzzy classification. The authors have used three steps for the detection of NTL. The first step comprises of assigning three different scores to each consumer using fuzzy clustering. The second score represents the change in the usage trend for the consumer. This change is recorded using membership matrices. The third score represents the deviation of the monthly consumption from the expected consumption. If the difference between the expected and real consumption profile passes a specific threshold, the consumer is shortlisted as a potential fraudster and an on-site inspection is recommended for a possible NTL detection. This work is performed on a small dataset of an electric supplier in Turkey. However, the paper does not use any of the known performance evaluation metrics.

#### 2.1.1.2 Additional Data Profile

Some authors have also tested merging consumption data with some additional data like data related to environment and temperature. For example, in (Hartmann et al., 2015), the authors have merged weather data with different consumption profiles which are based on time and type of customers (i.e., residential or industrial). This work is performed on a dataset collected from Creos Luxembourg, the electricity operator in Luxembourg. The authors have created multiple consumption profiles for each customer based on time, e.g, monthly, weekly and daily profiles and used them in live machine learning for consumer classification. Based on probability distribution and confidence rate, if a customer's consumption value surpasses the threshold, the system generates an alert for a possible NTL detection. The authors have claimed that the additional data coupled with live machine learning and maintaining multiple consumer profiles has helped reduce false positive rate (FPR). The results are evaluated using accuracy, precision, recall and $F1$ score.

### 2.1.2 Network-Based Techniques

Some interest has been developed in using network data to identify potential NTL. For example, Chauhan et al. (Chauhan, 2015) have proposed a framework to monitor current between poles. Given a constant voltage, the current between the poles will remain almost the same. If there is a large difference between the current readings of two poles, then it indicates that there is a possible unlawful connection between the poles. To some extent, this method can identify unlawful meter bypassing but it can not detect NTL which is caused by slowing down meters or wrong meter readings. The authors have not used any performance evaluation metric. A similar strategy is proposed in (Han and Xiao, 2014). The paper has proposed to install an observer meter for a community. A mathematical expression is also proposed, which calculates the difference between the billed amount of electricity and the total amount of consumed electricity. The observer meter is used to calculate the total electricity provided. This approach can be applied in AMI systems as well as in traditional grid systems. The authors have argued that this can help in detecting tempered meters from non-tempered meters. However, this solution can filter a locality where NTL is occurring but it fails to identify the specific consumers responsible for NTL.

In (Mutupe et al., 2017), the authors have proposed to remotely detect NTL by monitoring the difference between the electricity distributed and the electricity consumed. The electricity consumed at the consumer end is monitored by radio transceivers and communicated back to the distribution pole using Wi-Fi space. If the difference between the distributed and used electricity passes a certain threshold, a potential NTL is identified for an on-site inspection. This work is implemented in Eskom, the electric supply company in South Africa. However, this framework fails to identify NTL caused by meter bypassing. Moreover, a heavy cost is also associated with the installation of radio transceivers at every consumer's meter.

Another approach to detect NTL in neighborhood area for smart grids is discussed in (Xia et al., 2015). The authors have proposed a difference comparison-based inspection algorithm which uses binary inspection tree to calculate the difference in the amount of electricity stolen from a node to its child. The characteristics of binary search tree enables the algorithm to skip large amount of nodes which are useless to check. This helps in quickly identifying malicious meters. The algorithm keeps track of stolen electricity in the associated sub-tree of a node which helps in probing the next node.

### 2.1.3 Hybrid Techniques

Efforts have been made to combine the consumption data and network data in a bid to better achieve NTL detection results. In (Meira et al., 2017), the features are divided into four categories with respect to time, geography, similarity of consumption profile and infrastructure. Random forest, logistic regression and SVM are tested with different proportion of NTL ranging from 10% to 90% across all four categories. Results are compared using area under the curve (AUC). The results obtained from the consumption category are better than the results obtained from the infrastructure category. The paper concludes that predicting NTL using the raw data from consumption profile is better than using a combination of consumption and network data. The authors have also claimed that consumption downfall is not the only pattern of NTL rather an increasing consumption pattern can still be a good candidate for NTL. As AUC is the only metric used to evaluate performance of the classifiers, the relationships between different metrics can not be identified for NTL detection. In (Buevich et al., 2016), the authors have discussed two different techniques that separate NTL from the overall losses in an electric grid. One of them, the model-driven technique considers the examination of state of meters and the grid, packet losses and line losses. A large deviation of a specific meter in the regression indicates a possibility of NTL. The other technique, the data-driven, evaluates NTL using a classifier SVM on a synthetic data of different households. The authors have used true positive rate (TPR) and true negative rate (TNR) as the performance evaluation metrics. The authors have argued that the first technique helps to evaluate the grid, while the second technique gives an estimation of true positive rates (TPR) and true negative rates (TNR) with respect to different levels of NTL. The paper concluded that consumption data requires less configuration, which makes it relatively easier to implement.

Zhou et al. (Zhou et al., 2014) proposed a load profiling technique, which uses advantages of the two approaches for customer classification. One of the approaches is based on geographical location. Customers are grouped together on the basis of similar locality. The other approach is based on similar consumption patterns exhibited by the customers. These customers are then grouped in the same category. The authors have combined these two approaches to categorize customers on the basis of similar customers on similar region using the firefly algorithm (Yang et al., 2008) to detect NTL. They have performed experiments on the data collected from a power supply company in China. Accuracy is the only metric used to evaluate the performance. Thus, no comparison can be made with other metrics used for NTL detection.

## 2.2 Taxonomy and Summary of the Literature Review

Some of the contributions of the thesis is the derivation of a complete taxonomy for the strategies of NTL detection as described in Figure 2.2 and the literature review which is summarized in Table 2.1. The table contains the category, the referenced paper, the technique used, the performance evaluation metric used and limitations of the cited paper, if any.

TABLE 2.1: Summary of the Literature Review

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| Un-Sup. Learning | (León et al., 2011) | Association rule mining | support, confidence, TP, TN, FP and FN | The paper has not discussed about the strategy used for feature selection process |
| | (Singh and Yassine, 2018) | Association rule miningresults outperformed SVM and MLP | | |
| | (Sánchez-Zuleta, Fernández-Gutiérrez, and Piedrahita-Escobar, 2017) | Benford curve, hierarchical clustering and MDS | | |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| | (Sharma et al., 2017) | LOF, DBSCAN | Silhouette coefficient and Davies Bouldin index | has not compared results of metrics with other clustering algorithms |
| | (Zheng et al., 2017) | Distance matrix, | AUC, accuracy and F1 measures | |
| | (Avila, Figueroa, and Chu, 2018) | Signal processing technique, a smoothing spline function, under-sampling and outlier detection | Seven performance metrics | |
| | (Sharma and Singh, 2015) | DBMSCAN | Silhouette coefficient | |
| | (Júnior et al., 2016) | OPF, anomaly detection | Accuracy, F-Measure and standard deviation | has not discussed any feature selection technique |
| | (Yeckle and Tang, 2018) | Seven different outlier detection techniques | AUC | |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| | (Ramos et al., 2016) | BHA | Mean accuracy rates | has not discussed feature relevance of each feature |
| | (Ramos et al., 2011b) | Harmony search and OPF | Accuracy | Accuracy for Imabalanced dataset is not a good metric |
| | (Ramos et al., 2011a) | Harmony search, PSO and gravitational search | | The authors have not described the details of classifiers and their performance evaluation for NTL detection |
| Sup. Learning | (Zheng et al., 2018) | Wide and deep CNN | AUC | |
| | (Ford, Siraj, and Eberle, 2014) | ANN | TP, TN, FP and FN | |
| | (Cody, Ford, and Siraj, 2015) | M5P decision tree | RMSE | |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| | (Coma-Puig et al., 2016) | Naive Bayes, KNN, decision trees, neural networks, SVM, random forest and AdaBoost | Precision | |
| | (Bhat et al., 2016) | CNN, auto encoders and long short-term memory networks | precision, recall, F1 and ROC | |
| | (Chatterjee et al., 2017) | RNN | Accuracy | Accuracy for Imabalanced dataset is not a good metric |
| | (Spirić, Stanković, and Dočić, 2018) | Fuzzy logic | TP | Only percentage success is the metric used |
| | (Viegas, Esteves, and Vieira, 2018) | Fuzzy logic | TPR and FPR | |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| | (Glauner et al., 2016a) | Boolean rules, fuzzy logic and SVM | TP, TN, FP, FN, recall and specificity | The relationships that may exist between these metrics regarding NTL detection are not sufficiently discussed. |
| | (Punmiya and Choe, 2019) | CatBoost, LGBoost and XGBoost | FPR | The dataset is synthesized with equal positive and negative class representations |
| | (Di Martino et al., 2012) | Ensemble of one-class SVM, CS-SVM, OPF and C4.5 | accuracy, recall, precision and *F* value | Has not discussed the impact of using these metrics for NTL detection |
| | (Figueroa et al., 2017) | Random oversampling and undersampling, linear SVM, non-linear SVM and MLP | accuracy, recall, precision, specificity, AUC, $F_\beta$, $F_1$ and MCC | Has not discussed any feature selection strategy |
| | (Nagi et al., 2010) | SVM | Hit ratio | |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| Semi-Sup. Learning | (Júnior et al., 2016) | OPF, anomaly detection | Accuracy, F-Measure and standard deviation | has not discussed any feature selection technique |
| Hybrid of Supervised and Unsupervised Learning | (Guerrero et al., 2018) | ANN, regression tree and SOM | Accuracy | |
| | (Peng et al., 2016) | K-means clustering with DT, RF, SVM and KNN | Load curves, cluster evaluation indices and computing time | |
| | (Terciyanli et al., 2017) | Fuzzy clustering and fuzzy classification | No metric is used | The results are not evaluated using any known performance evaluation metric |

TABLE 2.1: Summary of the Literature Review (Cont.)

| **Category** | **Articles** | **Classifiers / Strategy** | **Eval. Measures** | **Limitations** |
|---|---|---|---|---|
| Add. Data Profile | (Hartmann et al., 2015) | Live machine learning on merged data of electricity and weather | Probability distribution, confidence rate, FPR, precision, recall and *F*1 | |
| Network-Based Techniques | (Chauhan, 2015) | Monitoring of current flow between poles | No metric is used | NTL caused by slowing down meters or wrong meter reading can not be identified |
| | (Han and Xiao, 2014) | Installation of observer meter | | It can filter the locality of NTL but fails to pinpoint specific consumer |
| | (Mutupe et al., 2017) | Installation of radio transceivers and Wi-Fi space | | IT can not detect NTL caused by meter by-passing. Heavy cost is also associated |
| | (Xia et al., 2015) | Binary inspection tree | Comparison of inspection steps | |
| Hybrid Techniques | (Meira et al., 2017) | RF, logistic regression and network data | AUC curve | The relationships between different metrics can not be identified for NTL detection |

TABLE 2.1: Summary of the Literature Review (Cont.)

| Category | Articles | Classifiers / Strategy | Eval. Measures | Limitations |
|---|---|---|---|---|
| | (Buevich et al., 2016) | State examination of meters with SVM | TPR and TNR | |
| | (Zhou et al., 2014) | Firefly algorithm uses geographical information and consumption profile | Accuracy | No comparison can be made with other metrics used for NTL detection |

### 2.2.1 Limitations

There have been many attempts to bring down NTL in different companies, regions and countries. A good success in identifying NTL is achieved by applying different machine learning classifiers. Different performance metrics are used to evaluate how good or bad the classifier is in predicting NTL. However, not much has been discussed about the relationships that exist between these performance metrics with respect to NTL. There is still a need to highlight performance evaluation metrics that are specifically suitable in evaluating machine learning classifiers for NTL problem.

One of the most important ingredients in solving a machine learning problem is the data used to solve the problem. Not only relevant records are important for correctly predicting the class labels but identifying relevant features are equally important. For NTL detection, efforts have been made in identifying and representing relevant records by addressing the issue of class imbalance but no such focus has been made in identifying relevant features. There is a

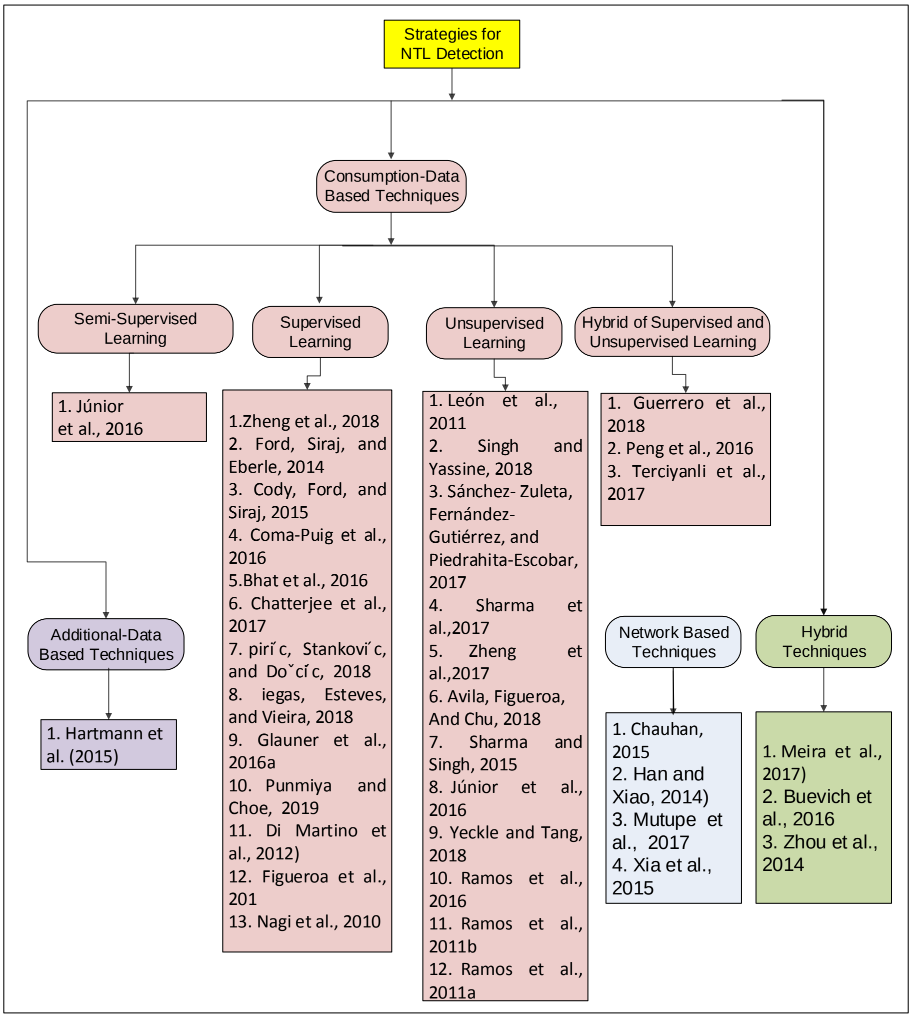

FIGURE 2.2: Taxonomy for strategies of NTL detection

need to develop a strategy that can systematically identify and select relevant features in the dataset before it is used for NTL detection.

A detailed comparative study of the machine learning classifiers on some real dataset for NTL detection is still missing. It should not only cover the impact of using the real dataset but it should also contain comparative results on the performance of different metrics using different machine learning classifiers.

### 2.2.2 Terms and Definitions

This section contains a detailed theoretical description of all the classifiers and the performance evaluation metrics used in this thesis.

#### 2.2.2.1 Naive Bayes Classifiers

All classifiers belonging to this type use Bayes' algorithm along with a 'Naive' assumption regarding the class conditional independence. The definition of Bayes' theorem as described in (Rish et al., 2001) is as follows:

$$P(C=i|X=x)=\frac{P(X=x|C=i)P(C=i)}{P(X=x)} \tag{2.1}$$

where $P(C=i|X=x)$ is the class posterior probability given the feature vector $X$. Notice that $P(X=x)$ is the same for all classes and thus can be ignored. Thus, Equation 2.1 is reduced to the following equation:

$$P(C=i|X=x)=P(X=x|C=i)P(C=i) \tag{2.2}$$

When the number of features is too many, computing $P(X=x|C=i)$ becomes exponentially expensive. For this reason, assumption of the class conditional independence is made. This assumption states that the features are independent of each other, which means that the values of one feature is not dependent on the values of any other feature. This assumption simplifies the Equation 2.2 to the following equation:

$$P(C = i|X = x) = \prod_{j=1}^{n} P(X_j = x_j|C = i)P(C = i) \tag{2.3}$$

Although this assumption is over-simplified, Naive Bayes (NB) still performs better on many real datasets. Zhang et al. (Zhang, 2004a) have explained why NB performs better with such an over-simplified assumption. They have argued that for a dataset with features having a high class conditional dependencies, NB can still perform better as long as the dependencies are well distributed among classes or if they cancel out each other. Different classifiers use NB approach, while the difference between them is the assumption they use for finding the posterior probability $P(X_j = x_j|C = i)$, where $x_j$ is the $j^{th}$ feature and $C = i$ is the $i^{th}$ class label. We have used Gaussian Naive Bayes (John and Langley, 1995) and Bernoulli Naive Bayes (McCallum, Nigam, et al., 1998) classifiers in our simulation.

#### 2.2.2.2 LDA and QDA

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) (Srivastava, Gupta, and Frigyik, 2007), (Balakrishnama and Ganapathiraju, 1998) belong to a separate type of supervised machine learning classifiers. As their names suggest, LDA generates linear decision boundaries and QDA generates quadratic decision boundaries. These classifiers are used in practice due to their advantage of multi-class support, computationally less expensive with no requirement of hyper-parameter tuning.

The difference between the two classifiers is that LDA uses the same co-variance matrix for all classes while QDA computes separate co-variance matrix for each class. Thus, at one hand, QDA is computationally expensive as compared to LDA but on the other hand it is more flexible and informative with respect to decision boundaries. In general, LDA works better with a small training set and thus has a low variance, while QDA performs better with a large training set and thus has a high variance.

#### 2.2.2.3 Generalized Linear Models

In this type, we have chosen logistic regression to simulate on our data. Unlike its name, logistic regression is used for classification instead of regression. The mathematical notation for logistic regression model (Ng, 2004) is given in Equation 2.4.

$$\hat{y}(w,x) = w_0 + w_1x_1 + ... + w_nx_n \tag{2.4}$$

where $\hat{y}$ is the predicted value, $x = (x_1, x_2, ..., x_n)$ is the feature vector, $w = (w_1, w_2, ..., w_n)$ is the coefficient vector and $w_0$ is the intercept.

For a binary classification, the value of $\hat{y}$ must be between 0 and 1. For this, a conversion function sigmoid is used. The mathematical notation of sigmoid is given in Equation 2.5.

$$s = \sigma(w_0 + w_1x_1 + ... + w_nx_n) = \sigma(z) = \frac{1}{1+e^{-z}} \tag{2.5}$$

where $s$ is the sigmoid function. For a large positive value of $z$, $s = 1$ and for a small or large negative value of $z$, $s = 0$ and for $z = 0$, $s = 0.5$. The objective of logistic regression is to minimize the error between actual and predicted values. To quantify this error, a loss function is used, which is further regularized by different regularizing schemes in order to minimize the error. Detailed analysis on regularizing schemes can be found in (Ng, 2004).

#### 2.2.2.4 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) (Zhang, 2004b) is a type of linear models that has a support of classification as well as regression. SGD is particularly attractive for problems having large number of observations and large number of features. Despite dealing with large data and high dimensionality, SDC is efficient and offers many options for tuning parameters like number of iterations and regularization parameters. However, one of the prerequisites of using SGD is that the data must be normalized before use which means it is sensitive to scaling. We have used SGD classifier in our simulation, which offers a lot of options for loss functions and their penalties. A training sample looks like $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$, where

$x_i \in \mathbf{R}^m$ and $y_i \in \{-1, 1\}$. Let us take a linear function which we want to learn

$$f(x) = w^T x + b \tag{2.6}$$

where $w \in \mathbf{R}^m$ and $b \in \mathbf{R}$ is the intercept. The training error, as described in (Zou and Hastie, 2005), is evaluated using Equation 2.7:

$$E(w, b) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \alpha R(w) \tag{2.7}$$

where $L$ is the loss function that estimates the difference between the expected and the actual output, $R$ is a regularization step used to penalize on occurrence of error and $\alpha > 0$ is a hyper-parameter. The two most commonly used regularization choices are $L1$ and $L2$ regularizations, given in Equations 2.8 and 2.9:

$$L1\ Regularization = \sum_{i=1}^{n} |w_i| \tag{2.8}$$

$$L2\ Regularization = \frac{1}{2} \sum_{i=1}^{n} w_i^2 \tag{2.9}$$

#### 2.2.2.5 Support Vector Machine

Vapinik has proposed the Support Vector Machine (SVM) classifier (Hearst et al., 1998) that creates a margin between the two classes and tries to maximize this margin. This type of classifier is a set of machine learning methods which offers support for outlier detection, regression and classification. SVMs are widely used in the field of data mining (Awais, Badruddin, and Drieberg, 2017; Raza et al., 2019) due to their high predicting power and reliability in supervised machine learning problems. The main strengths of SVM are its effectiveness on high dimensional data and on datasets where the number of features is greater than the number of observations, less memory consumption due to the use of support vector (which is a subset of training observations and not the whole training set) and the use of a variety of kernel functions which are used in the decision function. However, for a dataset where the difference between the number of features and number of observations is too big, SVM tends to overfit the model. Another associated disadvantage is that probability estimates are not calculated by SVM directly rather they can be calculated by some other costly operations.

SVM constructs an optimal decision function $f(x)$ that can predict unseen instances with high accuracy as given in Equation 2.10 where $sgn(g(x))$ is the boundary between the positive and negative classes (Vapnik, 1999).

$$f(x) = sgn(g(x)) \tag{2.10}$$

The expected error in classification is calculated using Expression 2.11 where $R$ is the expected error, $t$ is the training errors, $n$ is the number of training samples, $h$ is the dimension of the set of hyperplanes and $\eta$ is the confidence metric (Vapnik, 1998).

$$R < \frac{t}{n} + \sqrt{\frac{h\left(\ln\left(\frac{2n}{h}\right)+1\right) - \ln\left(\frac{\eta}{4}\right)}{n}} \tag{2.11}$$

SVM needs a comparatively smaller number of training samples. Therefore, unlike neural networks (Cao and Tay, 2003), it is less prone to getting struck with the problem of overfitting. Mapping of input to higher dimensions requires setting up of kernel for only a few thousands of training samples (Chang and Lin, 2011). This is a major concern in dealing with big datasets. To overcome this problem, linear SVC (Pedregosa et al., 2011a) uses linear kernel settings to make the data linearly separable. We have used linear SVC in our simulations.

#### 2.2.2.6 Decision Trees

Decision Trees (DT) are a set of machine learning methods used in classification and regression. The data provided to the decision tree is used to infer if-then-else rules. These rules become complex with the increase in depth of the tree. The strength of decision trees include its simple interpretation of the rules, no requirement of data normalization, computationally less expensive and handling of numerical as well as categorical data. However, its weaknesses include creation of over-complex tree in some cases resulting in overfitting, instability of the tree when new data is added and the problem of NP-completeness for an optimal decision tree. Different flavors of decision trees are available like $ID3, C4.5, C5.0$ and $CART$. These algorithms differ in handling categorical and numerical data along with pruning of if-then-else rules (Tsang et al., 2011).

#### 2.2.2.7 Neural Network Model

The Neural Network (NN) model offers Multi-layer Perceptron (MLP) classifier which is vastly used for classification. The objective of MLP classifier is to learn a function $f(.) : R^m \rightarrow R^n$ with a feature set $X = x_1, x_2, ...x_m$ and an output $y$ where $m$ is the number of features and $n$ is the number of values for the output $y$. The difference between MLP classifier and logistic regression is that MLP classifier can have one or more hidden layers between input and output layers. The input layer transforms the input to the hidden layers where a linear summation like $w_1x_1 + w_2x_2 + ... + w_mx_n$ occurs. The output layer takes the input from the hidden layer and converts it to the output values using the sigmoid function. The main advantage of using the MLP classifier is its compatibility with non-linear models. However, it requires a number of hyper-parameters to be tuned (Rumelhart, Hinton, Williams, et al., 1988).

#### 2.2.2.8 Nearest Neighbors

This type belongs to a set of supervised and unsupervised machine learning methods which are based on calculating distances from the neighbors (Goldberger et al., 2005). This technique is widely used in solving many real-world problems like physical activity classification (Awais, Palmerini, and Chiari, 2016). In nearest neighbors, both classification and regression are supported. The key idea of nearest neighbors is to find a predefined $(k)$ number of training observations which are closest to the new observation and then find the value of the output variable $y$ for this new observation based on the nearest neighbors. There are many metrics used to find the distance between neighbors. The most commonly used metric is standard Euclidean method. Generally, the success ratio of nearest neighbors is high for classification problems having irregular decision boundaries.

The main functionality of this classifier works slightly different than other learning techniques. KNN does not use the parameter of weight rather it is a record-based approach which uses $k$ nearest training samples to predict the value of the target variable.

Let $p = (p_1, p_2, \cdots, p_n)$ and $q = (q_1, q_2, \cdots, q_n)$ be the two samples.

The distance between the two samples is calculated using Equation 2.12.

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} \tag{2.12}$$

A test sample is assigned the class which is the most frequent in the $K$-nearest samples. The disadvantage associated with KNN is that as it is a record-based learning procedure, with the increase of k the time required to predict a test sample increases. The advantage of using KNN is that it does not depend on any other element, thus the runtime for prediction can be decreased by allocating different cores or nodes for parallel execution (Altman, 1992).

#### 2.2.2.9 Ensemble Methods

Ensemble methods combine the predicted results of multiple base estimators. This way the results are improved as compared to some individual estimator. There are two main streams of ensemble methods. The first stream includes techniques which take into account results from many individual estimators and combine their results using average. This way the combined results of individual estimators turn out to be better as compared to the results of individual estimators. Examples of this stream are bagging and random forests. The second stream includes techniques which combine many weak estimators in order to get a powerful result of an ensemble. This in turn also reduces the bias. Examples of this stream are AdaBoost, CatBoost, LightGBM and XGBoost. To test the behaviors of both streams, we have experimented with random forest, AdaBoost, CatBoost, LightGBM and XGBoost in our simulations.

AdaBoost is a technique to repeatedly apply new data to weak estimators (Freund and Schapire, 1997). This includes increasing weights for the training observations that had wrong predictions and decreasing weights for the training observations that had correct predictions. This way, with every new iteration, the estimator is restricted to concentrate on those training observations that had wrong predictions in the previous iteration.

A random forest comprises of multiple individual decision trees (Liaw, Wiener, et al., 2002a). For each tree, a separate set of training examples is selected. Using this approach, the problem of over fitting in imbalance datasets is avoided. On the testing phase, the final outcome of a sample is evaluated by using the majority voting scheme from among all the individual decision trees. Another advantage of using this approach is that as different training examples are used in every decision tree, variable number of nodes or cores can be used for training (Ho, 1995).

Derived from the terms 'Category' and 'Boosting', CatBoost is an open-source machine learning algorithm (Prokhorenkova et al., 2018). The term 'Category' refers to the fact that it handles categorical features on its own. Other machine learning techniques require pre-processing steps to convert categorical data into numbers but CatBoost requires only the indices of categorical features. It then automatically performs one-hot encoding to transform the categorical data into numerical data (Dorogush, Ershov, and Gulin, 2018). The term 'Boost' refers to the fact that it is based on gradient boosting algorithm which itself is widely used in different machine learning problems like recommendation systems, fraud detection, forecasting, etc. Moreover, unlike deep learning, CatBoost does not require huge datasets for extensive training. Despite having a number of hyper-parameters like regularization, learning rate, number of trees, tree depth, etc., CatBoost does not require exhaustive hyper-parameter tuning which reduces the likelihood of overfitting.

CatBoost uses three steps to transform categorical features having a number of categories greater than a specified number into numerical features.

1. The set of input observations are randomly permuted multiple number of times.

2. The label values are transformed from categorical or floating point to integer values.

3. The categorical features are transformed to numerical features using the formula given in Equation 2.13:

$$Average\_target = \frac{InClassCounter + Prior}{TotalCounter + 1} \tag{2.13}$$

where *InClassCounter* represents the number of times the class label is 1 for all those records having the current feature value. *Prior* is the starting value for the numerator and is defined during initialization of parameters. *TotalCounter* is the total number of records (up to the previous record) having the same categorical value as that of the current categorical value.

Suppose *Class-Bill* is a feature that contains categorical values representing the category of the consumer. The feature can contain three categories namely home, industrial or government. Table 2.2 shows the observations after applying random permutation. Table 2.3 shows the transformed categorical values of *Class-Bill* into numerical values using Equation 2.13. In this case, we have set *Prior* to 0.05.

TABLE 2.2: Observations after applying random permutation

| Records | Class-Bill | Class Label |
|---|---|---|
| 1 | Home | 0 |
| 2 | Industrial | 0 |
| 3 | Home | 1 |
| 4 | Home | 1 |
| 5 | Government | 1 |
| 6 | Industrial | 1 |
| 7 | Home | 0 |

Light Gradient Boosting Machine (LGBoost) is another gradient-based boosting algorithm which uses decision trees (Ke et al., 2017). Like CatBoost, it is used in many machine learning problems involving classification and prediction. Instead of level-by-level tree

TABLE 2.3: Observations after transforming categorical data into numerical values

| Records | Class-Bill | Class Label |
|---|---|---|
| 1 | 0.05 | 0 |
| 2 | 0.05 | 0 |
| 3 | 0.025 | 1 |
| 4 | 0.35 | 1 |
| 5 | 0.05 | 1 |
| 6 | 0.025 | 1 |
| 7 | 0.5125 | 0 |

growth, LightGBM uses depth-first approach in splitting the tree which may cause an increase in complexity and overfitting. To avoid this disadvantage, maximum depth of the tree can also be set. The training time of LightGBM is significantly improved as it converts continuous feature values into discrete bins using a histogram approach. However, it is not advisable to use LightGBM for smaller datasets as it tends to overfit them.

Extreme Gradient Boost (XGBoost) (Chen and Guestrin, 2016) is another boosting-based machine learning algorithm. Unlike CatBoost, it does not transform categorical data into numbers by its own. Consequently, before applying XGBoost, a pre-processing step must include encoding techniques like one-hot encoding, mean encoding or label encoding to convert categorical features into numerical features. It also has a built-in feature of handling missing values. A specific parameter is reserved to supply a different value than the usual values which is used by the algorithm when it encounters a missing value.

Table 2.4 shows a comprehensive comparison between KNN, SVM and random forest. These three classifiers, used in Chapter 4, are compared on the basis of the number of training samples, the usage of distance function, hyper plane or decision trees, and parallel processing.

### 2.2.3 Evaluation Metrics

This section contains a detailed theoretical description and formulae of all the performance evaluation metrics used in this thesis.

TABLE 2.4: Comparison of KNN, SVM and random forest

| | KNN | SVM | Random forest |
|---|---|---|---|
| No. of training samples | Very Small | Small | Large |
| Uses distance function | $\surd$ | $\times$ | $\times$ |
| Construct a hyper plane | $\times$ | $\surd$ | $\times$ |
| Uses decision trees | $\times$ | $\times$ | $\surd$ |
| Occurrence of over fitting | $\surd$ | $\times$ | $\times$ |
| Parallel processing | $\surd$ | $\times$ | $\surd$ |

One of the most common performance evaluation metric used for classifiers is accuracy. It gives a measure that how accurately the classifier has predicted TP and TN. It tends to get failed for imbalanced datasets where the user preference is towards the FP and FN. For example, talking about NTL, 99% of electric consumption is a normal usage and only 1% of consumption is a theft case. Now if a classifier correctly predicts all 99% of normal usage and does not predict the remaining 1% of theft usage, accuracy will be measured as 99%. In reality, the classifier was not performing well because it failed to predict the theft class. Equation 2.14 is used to calculate accuracy.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.14}$$

Due to lack of handling measure of predicting FP and FN, other measures are derived which either take care of both classes separately or handle the least represented class more accurately. These are discussed through Equations 2.15-2.20.

$$TruePositiveRate(Recall) = \frac{TP}{TP + FN} \tag{2.15}$$

$$TrueNegativeRate(Specificity) = \frac{TN}{TN + FP} \tag{2.16}$$

$$FalsePositiveRate = \frac{FP}{TN + FP} \tag{2.17}$$

$$FalseNegativeRate = \frac{FN}{FN + TP} \tag{2.18}$$

$$PositivePredictiveValue(PPV)orPrecision = \frac{TP}{TP + FP} \tag{2.19}$$

$$NegativePredictiveValue(NPV) = \frac{TN}{TN + FN} \tag{2.20}$$

True Positive Rate (TPR) or recall is the measure of total number of thieves correctly classified as thieves by the classifier. Recall is also called as sensitivity. As we want to minimize FN, by Equation 2.15 minimizing FN will maximize recall. This gives an indication that NTL requires a high recall model. The higher the recall, the better it is for NTL. On the other hand, by equation 2.19 we see that if precision is low, the model can still tolerate because it does not need a high precision. True Negative Rate (TNR) is also called as specificity. It is a measure that shows how many of the total negative instances were correctly classified as negative. False Positive Rate (FPR) is the measure of total number of normal consumers wrongly predicted as thieves. False Negative Rate (FNR) is a measure that out of total positive instances how many were wrongly classified as negative. Positive Predictive Value (PPV) or precision is the measure that out of the total predicted positive class instances how many were classified correctly as positive. Negative Predictive Value (NPV) is the measure that out of total predicted negative class instances how many were correctly classified as negative.

$$F_\beta = \frac{\left(1 + \beta^2\right) Recall \times Precision}{\beta^2 \times Precision + Recall} \tag{2.21}$$

$F_\beta$, as shown in 2.21, is another metric that is used for evaluation of classifiers in imbalance datasets ("A Survey of Predictive Modeling on Imbalanced Domains"). It uses recall (completeness) and precision (exactness) where $\beta$ is a coefficient used to set the priority between recall and precision. For $\beta = 1$, both recall and precision have the same priority. If $\beta$ is set to a value greater than 1, recall gets the high weightage and if it is set to a value smaller than 1, precision gets the high weightage. Usually people use value 1 when dealing with imbalance domain. We have tested two different values of $\beta$, i.e. with $\beta = 1$ and with $\beta = 1.5$. The latter case sets the priority of recall higher than precision. When both recall and precision are high, the value for $F_\beta$ becomes high.

$$ArithmeticMean = \frac{(Precision + Recall)}{2} \tag{2.22}$$

$$HarmonicMean = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.23}$$

$$G - Mean = \sqrt{Sensitivity \times Specificity} \tag{2.24}$$

Arithmetic mean is the average of precision and recall measure as shown in Equation 2.22. It is rarely used as evaluation metric for imbalance datasets as it does not give an insight to the two performance measures. Instead, harmonic mean is preferred which is presented in Equation 2.23. Seeing the equation, it is obvious that it is always less than the arithmetic mean of the two. In fact, harmonic mean is closer to the smaller of the two values. So if harmonic mean is high, that is an indication that both precision and recall are high (Sun et al., 2007). $F_\beta = 1$ is the harmonic mean of precision and recall. Geometric mean (G-Mean), presented in Equation 2.24, is used when performance measure of both TPR and TNR is of concern. It is a measure that how good the classifier is for both classes.

$$Dominance = TPR - TNR \tag{2.25}$$

In (García, Mollineda, and Sánchez, 2008), a new performance measure, dominance, was proposed. It gives a measure of dominance between the positive and the negative classes. Seeing Equation 2.25, it can be deduced that it ranges between $-1$ and $+1$. A value close to $+1$ indicates good accuracy of the classifier for the positive class and a value close to $-1$ depicts good accuracy of the classifier for the negative class.

### 2.2.4 Summary

In this chapter, we discussed a thorough literature survey in NTL detection. The activities performed in recent years were highlighted and summarized in a summary table. Moreover, a complete taxonomy for the strategies of NTL detection was also presented. Moreover, it highlighted the limitations identified in NTL detection. Finally, the chapter included the theoretical description and the necessary mathematical representation of all the classifiers and the performance evaluation metrics used in this thesis.

# Chapter 3

# Data Collection and Pre-Processing

## 3.1 Proposed Methodology

A set of certain pre-processing steps are required in order to transform the raw data into a form suitable for data analytics. This includes data collection, data munging, feature selection, data merging and feature scaling. In the following sub-sections, a description of every step can be found. The methodology is presented in Figure 3.1.

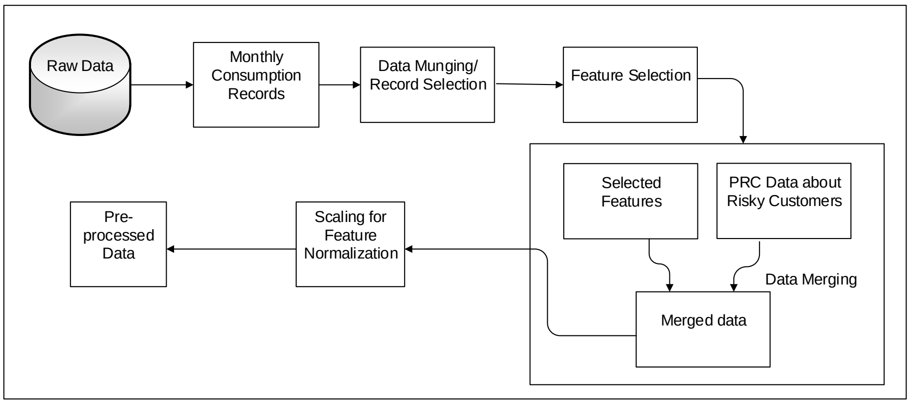
Raw Data
Monthly Consumption Records
Data Munging/ Record Selection
Feature Selection
Selected Features
PRC Data about Risky Customers
Data Merging
Merged data
Scaling for Feature Normalization
Pre-processed Data

FIGURE 3.1: Proposed methodology for NTL detection

### 3.1.1 Data Collection

Relevant data is the most important ingredient for a success or a failure of a solution to a problem. Percentage of success is increased with the increase of relevancy of data. For NTL detection problem, a major constraint is access to real data. For a power supply company, the client's data must be secured. For this reason, no real data is freely available from power supply companies. For NTL detection, access to real data becomes more important due to the fact that only real data can give real insights to the occurrences of NTL. On the other hand, synthesized data has its own limitations as it does not represent real instances. There have been many attempts for NTL detection in synthesized data. The techniques used to synthetically produce instances of NTL include decreasing the units consumed or decreasing the amount billed. However, in real-life, many factors affect the occurrences of NTL like the neighborhood area, amount of load shedding, the billed amount, etc. These factors are ignored in synthetically built data.

We have collected a dataset from an electric supply company in Pakistan. The collected data contains monthly consumption records of consumers which range between January 2015 and March 2016. It comprises of $80,244$ monthly consumption records. For privacy reasons, we have changed the feature names and have omitted the clients' details. The dataset is randomly split into train and test sets with the ratio of 80% and 20%, respectively. The training set contains $64,195$ records out of which $61,456$ are normal records with no theft and $2,739$ are abnormal records where the users have committed stealing of electricity. The test set contains $16,049$ records out of which $15,366$ are normal consumption records and 683 records contain NTL. The percentage of NTL in both sets is 4%.

As the number of normal users is always much greater than the number of abnormal users (thieves), this data is considered imbalanced and biased towards major representation of normal users. This behavior is shown in Figure 3.2 and a detailed characteristic chart of train and test data is presented in Table 3.1.

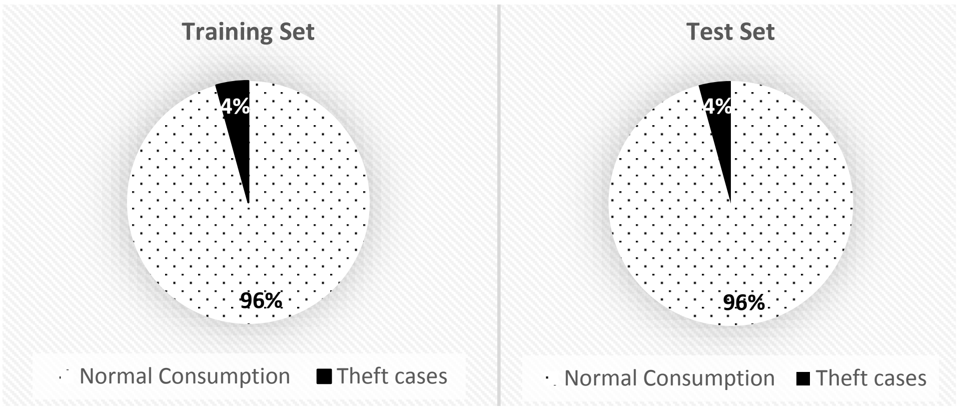

FIGURE 3.2: Imbalanced Data Representation

TABLE 3.1: Data Characteristic Chart

| Sr. No | Parameter | Unit |
|---|---|---|
| 1 | Number of observations | 80244 |
| 2 | Number of features | 71 |
| 3 | Train percent split | 80% |
| 4 | Test percent split | 20% |
| 5 | Train size | 64195 |
| 6 | Test size | 16049 |
| 7 | Normal consumption in Train set | 61456 |
| 8 | Theft cases in Train set | 2739 |
| 9 | Normal consumption in Test set | 15366 |
| 10 | Theft cases in Test set | 683 |
| 11 | Percentage theft in Train set | 4.45% |
| 12 | Percentage theft in Test set | 4.44 |

### 3.1.2 Data Munging

Not every consumption record is useful in identifying NTL. Some records may contain many NULL entries while others may contain data which is not suitable for the identification of NTL. For this reason, a subset of records should be built before it is used in NTL detection. Data munging is the process which performs the creation of the subset.

The data obtained from the electric company is in a comma separated values (CSV) file. Initially, the raw data collected from the power supply company contained 110 features. Some of the features are redundant and useless. For example, the feature ′Postal Code′ contains the same code for all records, the feature ′Meter Number′ and ′Registration Number′ are both used for unique identification, and the feature ′Write-Off′ contains any relief of dues which actually contains all zero entries. After filtering out the useless features, the feature set is reduced to 71 features.

The selected features need further transformation from a raw format into a form suitable for downstream analytic processing. This includes conversion of string types to numerical types, conversion of Not-a-Number (NaN) type to numeric and conversion of date data type. The transformation of every feature value is shown in Table 3.2.

The feature 'Category-Rate' records different categories of bill like residential, commercial, industrial, etc. The string values are transformed to numbers ranging from 0 to 15. 'Date-Last Disconnection' records the dates on which a connection was disconnected. Any entry of date represents that the connection was disconnected at some time. So, all the date entries are transformed to 1 and NaN values are transformed to 0. The feature 'Last Discon Reason' stores the reason disconnection was performed. This could be due to nonpayment of bill, empty house or NTL detection. NaN's are converted to 0. The feature 'Class-Bill' records the bill class according to the type of customers. The string values and NaNs are transformed to numeric values. The feature 'DC Ordinary IP' records the details of direct current (DC) to industries or normal users while 'Ordinary IP' records the details of alternate current (AC). The string values are transformed to numeric values. The feature 'Conn Phase' records the connection phase granted like single or three phase etc. Their corresponding string values are transformed to numeric values. The feature ‘Bill Day’ records the day of month the reading was taken. The original values are retained with no transformation. The features ‘Meter Reading Unit’, ‘Load Sanctioned’ and ‘Load-Connection’ record units consumed, load allowed and load used, respectively. Original values are retained for these three features. The feature ’Status’ records current status of the connection which could be either of active, in-active or disconnected. The string values are transformed to numeric values.

The feature ‘Type-Premise’ records the type of premise in which the electricity connection is provided. A variety of premise types are found in the data, e.g. shop, house, flat, school, mosque, bank, etc. All the string values are converted into numeric values ranging from 0 to 42. The feature ‘Code-Set-Aside’ records the code for any disputed amount and the feature ‘Amount-Set Aside’ records the disputed amount. The disputed amount is unchanged while the string codes are transformed to numeric codes. The feature ‘InstallementNo’ stores the installment number whereas the feature ‘Amount - Installement’ stores the installment amount, if any. Only NaN values are transformed to 0 while other data is retained with no transformation being performed. The feature ‘Type-Bill’ records the type of bill, e.g. residential, commercial, etc. The string values are transformed to numeric values. The feature ‘Type-Consumer’ records the detail of consumer connection like single phase, 3 phase, low tension (LT) and high tension (HT). The string values are transformed to numeric values.

Class of industry is categorized in the feature ‘Industry Class’ with an LT, HT or normal connection. Their corresponding string values are transformed to numeric values. The feature ‘Date-Last Payment’ records the date last payment was made. The original dates are retained while NaN values are transformed to a default date of 1/1/2001. The feature ‘Amount-Last Payment’ records the last amount paid. The original values are retained while NaN values are transformed to 0. There are many manufacturers of meters which are recorded in the feature ‘Meter Company’. Their string codes are transformed to numeric values ranging from 1 to 49. A feature value with no meter information is recorded as 0. There are a total of 15 meter categories recorded in the feature 'Meter Category'. Their categories are transformed to numeric values. The information of Security Intelligence Report (SIR) is stored in features 'Last SIR No' and 'Last SIR Date'. This information is retained with no transformation. The feature 'Month-Billing' records billing month. No transformation is performed for this feature. The feature 'Category-DC Rate' records different rates applied to DC connection. These rates can be regular, revised or irregular. The string categories are transformed to numeric values.

The features 'Units-Normal', 'Units-Average', 'Amount-Average' record normal units consumed, average units consumed and average amount paid. Average units are recorded due to non-reading of meter. Original data is retained for these features. The feature 'Units-Adjusted' records some adjusted units from previous month and the feature 'Amount-Adjusted' records the adjusted amount from previous month. Original data is retained for these features. The feature 'Units-Regular' records the sum of all units while the feature 'Amount-Regular' records the payable amount for regular units. No transformation is performed for these features. The feature 'Units-Current' records a sum of regular and irregular units while 'Amount-Current' records the amount of current units. Original data is retained for these features. The feature 'Units-12MonthsAvg' records average of units consumed during the last 12 months while 'Amount-12MonthsAvg' records the amount based on average units consumed during the last 12 months. The feature 'BilledUnit-YTD' records total billed units in the current year. The feature 'Units-12Monts' records total units consumed during the last 12 months while the feature 'Amount-12Months' records the total amount paid during the last 12 months. Original data of all these features is retained. The feature 'Balance-Opening' records the credit amount when the bill was issued. The feature 'BilledAmount' records the bill of the current month. The feature 'Billed-LPS' records late payment surcharge. The feature 'Amount-LPSWaived' records the waived amount of late payment surcharge. The feature 'BankComm' records the commission charged by the bank. The feature 'Payment Received' records the received payment from the consumer. The feature 'Amount - Adjustement' records the adjusted amount for the next month.

Original data is retained for all these features. The feature 'Request-Installement' records installment request which is mostly 0. The feature 'Amount Clearing' records amount stored for clearing account. The NaN values are transformed to 0. The feature 'Balance-Closing' records the credit amount after a bill is issued. The NaN values are transformed to 0. Late payment surcharge is stored for different time period ranging from 1 month to 5 years. This is stored in their respective features like '1 Month LPS' and '$1 - 2MonthLPS$', etc. Original values are retained for these features while NaN values are transformed to 0. The feature 'Amount-GrossBilledYTD' records the total gross bill in current year while the feature 'Amount-NetCreditYTD' records the total credit of consumers in current year. The feature 'BilledAmount-12MonthsGross' records the total payment made by the consumer during 12 moths while the feature 'Amount-12Months NetCredit' records the total credit amount of customers during the last 12 months. Original values are retained for these features while NaN values are transformed to 0. The feature 'Amount-12MonthsAvg GrossBilled' records the average of gross amount for the last 12 months while the feature 'Amount-12MonthsAvgNetCredit' records the average of credit amount for last the 12 months.

The company has provided a separate detail of NTL occurrences of consumers. This data is merged with the available features. The merging process is described in Section 3.1.4. The NTL occurrence is measured with the help of a feature 'NTL'. This feature has 16 different identifications of consumers out of which three are categorized as thieves. These are 'IRB (THEFT)-IBC', 'Assessed' and 'ITG - Irregular Bill against Tariff Revi'. We have transformed these to 1 showing an NTL occurrence (Theft) while the remaining are transformed to 0 showing a normal consumption (No-Theft). The Python code for the pre-processing steps is shown in Appendix C.

TABLE 3.2: Transformation of Feature values

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| Category-Rate | NaN | 0 |
| | A1-R | 1 |
| | A2-C_RET | 2 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | A2-C | 3 |
| | B-2 | 4 |
| | B-1 | 5 |
| | A1-R_EM_BE | 6 |
| | E-1_I | 7 |
| | E-1_II | 8 |
| | A2-C_B | 9 |
| | B-3_TOD | 10 |
| | E-2_I | 11 |
| | A2-C_B_RET | 12 |
| | MIX | 13 |
| | D-1 | 14 |
| | B-2_TOD | 15 |
| Type-Bill | NaN | 0 |
| | A2-C | 3 |
| | A1-R | 1 |
| | B-2 | 4 |
| | E-1_II | 8 |
| | B-1 | 5 |
| | E-2_II_A | 16 |
| | E-2_I | 11 |
| | C-1 | 17 |
| | B-3 | 18 |
| Date-Last Disconnection | All dates | 1 |
| | NaN | 0 |
| Last Discon Reason | NaN | 0 |
| | 0 | 1 |
| | 3 | 2 |
| | 4 | 3 |
| Class-Bill | NaN | 0 |
| | A1-R | 1 |
| | A2-C | 2 |
| | B | 3 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | E | 4 |
| | MIX | 5 |
| | D | 6 |
| DC Ordinary IP | NaN | 0 |
| | ORD | 1 |
| | IND | 2 |
| Ordinary IP | NaN | 0 |
| | ORD | 1 |
| | IND | 2 |
| Conn Phase | NaN | 0 |
| | SINGLE | 1 |
| | POLY<20 | 2 |
| | POLY20TO90 | 3 |
| Bill Day | Original Data | None |
| Metre Reading Unit | Original Data | None |
| Load-Sanctioned | Original Data | None |
| Load-Connection | Original Data | None |
| Status | ACT | 1 |
| | DIS | 2 |
| | MOC | 3 |
| Type-Premise | NaN | 0 |
| | HOUSE | 1 |
| | SHOP | 2 |
| | FLAT | 3 |
| | INDUSTRY | 4 |
| | UNKNOWN | 5 |
| | SCHOOL | 6 |
| | TELECOM TOWER / PTCL | 7 |
| | MOSQUE | 8 |
| | BANK | 9 |
| | NOT AVAILABLE | 10 |
| | MARRIAGE HALL | 11 |
| | GODOWN | 12 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | HOSPITAL - PRIVATE | 13 |
| | CNG STATION | 14 |
| | PETROL PUMP | 15 |
| | OFFICE - PRIVATE / LAWYERS / SOLICITORS | 16 |
| | MADARSA | 17 |
| | RESTAURANT | 18 |
| | HOTEL | 19 |
| | DISPENSARY / CLINIC / LABORATORY -GOVT | 20 |
| | FACTORY | 21 |
| | SOFTWARE HOUSE | 22 |
| | SHOPS | 23 |
| | DISPENSARY / CLINIC / LABORATORY- PVT | 24 |
| | OFFICE - GOVT | 25 |
| | CHARITABLE INSTITUTE / NGO / WELFARE | 26 |
| | MOBILE TOWER | 27 |
| | TUBE WELL - FISH FARM, NURSERIES, FISH H | 28 |
| | IMAM BARGAH | 29 |
| | PARKS / PLAYGROUND | 30 |
| | HOSPITAL - GOVT | 31 |
| | GRAVEYARD | 32 |
| | NEON | 33 |
| | OFFICE | 34 |
| | STREET LIGHT | 35 |
| | GRID STATION | 36 |
| | WATER PUMP | 37 |
| | OFFICES | 38 |
| | HOSPITALS / DISPENSERIES | 39 |
| | COLLEGE | 40 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | POST OFFICE | 41 |
| | FISH HATCHERIES | 42 |
| Code-Set Aside | NaN | 0 |
| | 8 | 8 |
| | 7 | 7 |
| | Y | 1 |
| | 4 | 4 |
| | K | 2 |
| | J | 3 |
| | G | 5 |
| Amount-Set Aside | NaN | 0 |
| | Original Data | None |
| InstallementNo | NaN | 0 |
| | Original Data | None |
| Amount-Installement | NaN | 0 |
| | Original Data | None |
| Type-Consumer | NaN | 0 |
| | DOL Connection | 1 |
| | DOL Connection 3 phase (mostly) | 2 |
| | Bulk LT Connection | 3 |
| | HT Connection | 4 |
| Industry Class | NaN | 0 |
| | Small Industy | 1 |
| | Low Tension - Large Industry | 2 |
| | High Tension - Large Industry | 3 |
| Partner | NaN | 0 |
| | EVALUATION GRID PRIVATE LIMITED | 1 |
| Date-Last Payment | NaN | 1/1/2001 |
| | Original Data | None |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| Amount-Last Payment | NaN | 0 |
| | Original Data | None |
| Meter Company | NaN | 0 |
| | SBL | 1 |
| | PEL | 2 |
| | CHA | 3 |
| | EAC | 4 |
| | EPL | 5 |
| | DKB | 6 |
| | KRZ | 7 |
| | SPC | 8 |
| | USR | 9 |
| | TEC | 10 |
| | KOR | 11 |
| | DMT | 12 |
| | GNZ | 13 |
| | DTM | 14 |
| | DEP | 15 |
| | EEC | 16 |
| | CAH | 17 |
| | KBK | 18 |
| | INT | 19 |
| | MTI | 20 |
| | ISK | 21 |
| | Unknown | 22 |
| | IND | 23 |
| | DSB | 24 |
| | LAG | 25 |
| | ACT | 26 |
| | PAF | 27 |
| | SSW | 28 |
| | EMC | 29 |
| | CEL | 30 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | POL | 31 |
| | HTL | 32 |
| | PRI | 33 |
| | SRH | 34 |
| | SBS | 35 |
| | DPE | 36 |
| | SBE | 37 |
| | UHR | 38 |
| | ABB | 39 |
| | MPV | 40 |
| | BEM | 41 |
| | LG | 42 |
| | VER | 43 |
| | SPT | 44 |
| | AIS | 45 |
| | BIC | 46 |
| | CRE | 47 |
| | DIN | 48 |
| | ELS | 49 |
| Meter Category | NaN | 0 |
| | METER; ENERGY SINGLE PHASE 10-40A | 1 |
| | METER; STATIC SINGLE PHASE | 2 |
| | METER; ENERGY 3 PHASE 15-90A | 3 |
| | METER; STATIC THREE PHASE | 4 |
| | METER; 3PH DOL AMR GPRS/3G W/DC SW | 5 |
| | METER; HT CT/PT&LT OPERATED W/GPRS MODEM | 6 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | METER; STATIC LT C.T.O PROG. | 7 |
| | METER; ENERGY SINGLE PHASE (REFURBISHED) | 8 |
| | METER; 1PH DOL AMR GPRS/3G W/DC SW | 9 |
| | METER; ENERGY C.T OPERATED 100/5A | 10 |
| | METER; ENERGY THREE PHASE (REFURBISHED) | 11 |
| | METER;DIRECT ONLINE W/GPRS MODEM 3 PHASE | 12 |
| | METER E 42 F-D/M 400/5A, 66000/100 V | 13 |
| | USED METER E 42 F-DM 400/5A, 66000/100 V | 14 |
| | USED METER; STATIC LT C.T.O PROG. | 15 |
| Last SIR No | NaN<br>Original Data | 0<br>None |
| Last SIR Date | NaN<br>Original Data | 0<br>None |
| Month-Billing | Original Data | None |
| Category-DC Rate | Regular<br>IRB-Detection<br>IRB-Revised | 1<br>2<br>3 |
| Units-Normal | Original Data | None |
| Units-Average | Original Data | None |
| Amount-Average | Original Data | None |
| Units-Adjusted | Original Data | None |
| Amount-Adjusted | Original Data | None |
| Units-Regular | Original Data | None |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| Amount-Regular | Original Data | None |
| Units-Current | Original Data | None |
| Amount-Current | Original Data | None |
| Units-12MonthsAvg | Original Data | None |
| Amount-12MonthsAvg | Original Data | None |
| BilledUnit-YTD | Original Data | None |
| Units-12Months | Original Data | None |
| Amount-12Months | Original Data | None |
| Balance-Opening | Original Data | None |
| BilledAmount | Original Data | None |
| Billed-LPS | Original Data | None |
| Amount-LPSWaived | Original Data | None |
| BankComm | Original Data | None |
| Payment Received | Original Data | None |
| Amount-Adjustment | Original Data | None |
| Allowance-PreviousYear | NaN<br>Original Data | 0<br>None |
| Request-Installement | NaN<br>Original Data | 0<br>None |
| Amount-Clearing | NaN<br>Original Data | 0<br>None |
| Balance-Closing | NaN<br>Original Data | 0<br>None |
| Amount-OutStanding | NaN<br>Original Data | 0<br>None |
| 1YearorAboveLPS | NaN<br>Original Data | 0<br>None |
| 1 yearorLessLPS | NaN<br>Original Data | 0<br>None |
| 1 Month LPS | NaN | 0 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | Original Data | None |
| 1-2MonthLPS | NaN<br>Original Data | 0<br>None |
| 2-3MonthLPS | NaN<br>Original Data | 0<br>None |
| 3-6MonthLPS | NaN<br>Original Data | 0<br>None |
| 6-12MonthLPS | NaN<br>Original Data | 0<br>None |
| 1-2YearLPS | NaN<br>Original Data | 0<br>None |
| 2-3YearLPS | NaN<br>Original Data | 0<br>None |
| 3-4YearLPS | NaN<br>Original Data | 0<br>None |
| 4-5YearLPS | NaN<br>Original Data | 0<br>None |
| Above5YearLPS | NaN<br>Original Data | 0<br>None |
| Amount-GrossBilledYTD | NaN<br>Original Data | 0<br>None |
| Amount-NetCreditYTD | NaN<br>Original Data | 0<br>None |
| BilledAmount-12MonthsGross | NaN<br>Original Data | 0<br>None |
| Amount-12MonthsNetCredit | NaN<br>Original Data | 0<br>None |
| Amount-12MonthsAvgGrossBilled | NaN | 0 |

TABLE 3.2: Transformation of Feature values (cont.)

| Feature Name | Original Value | Trans. Value |
|---|---|---|
| | Original Data | None |
| Amount-12MonthsAvgNetCredit | NaN | 0 |
| | Original Data | None |
| NTL | IRB (THEFT) - IBC | 1 |
| | Normal | 0 |
| | Assessed | 1 |
| | Adjusted | 0 |
| | Average | 0 |
| | ITG - Irregular Bill against Tariff Revi | 1 |
| | IBC -Wrong Tariff | 0 |
| | RB (MC DISCREPANCY) - IBC | 0 |
| | IRB (OTHER) REVISION | 0 |
| | IBC - Assessed Bill Revision | 0 |
| | WRONG TARIFF | 0 |
| | WRONG READING / PUNCHING / POSTING / MMF / CALCU | 0 |
| | Extra GST and/or Further GST Revision | 0 |
| | MIDDLE BILL | 0 |

### 3.1.3 Feature Selection

From the set of 71 features, a subset of useful features can be shortlisted using feature importance. It is a measure of finding the importance of each feature (Breiman, 2001). A feature has an importance if the model's error of prediction is increased with the shuffling of the value of the feature. The increase in the model's prediction error indicates that the model relies on that feature. Thus, the feature is important. Conversely, a feature has less importance if the model's error of prediction is not changed with the shuffling of the value of the feature. The stability in the model's prediction error indicates that the model does not rely on that feature. Thus, the feature is not important. This way, we can not only find the optimum combination of features but it can also significantly reduce the computational time of the classifiers. The details of feature selection strategies used and their advantages are discussed separately in the contributions listed in Chapters 4, 5 and 6.

### 3.1.4 Data Merging

Additionally, the company also provided the data of potential risky consumers (PRC). These consumers are identified during on-site inspection. This data is useful in assigning the values of class labels as true or false. A class label of true indicates an instance of a theft and a class label of false indicates an instance of a normal consumption. This process is shown in Figure 3.3. The PRC data is merged with the data shortlisted from the feature selection module.

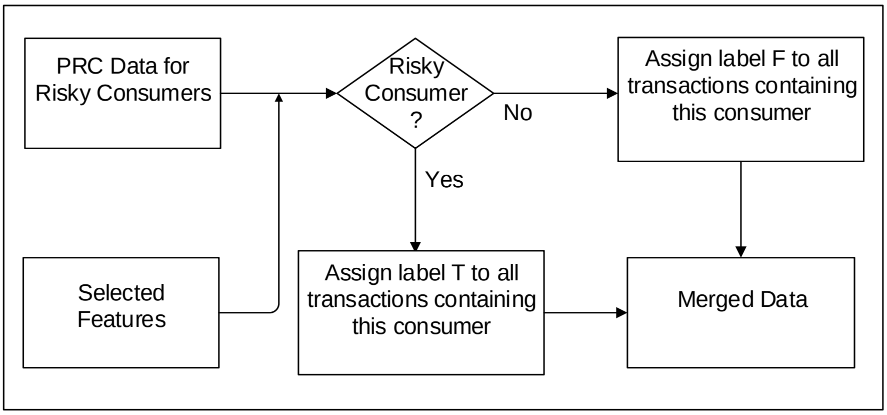
PRC Data for Risky Consumers
Selected Features
Risky Consumer ?
No
Yes
Assign label F to all transactions containing this consumer
Assign label T to all transactions containing this consumer
Merged Data

FIGURE 3.3: Data Merging

### 3.1.5 Feature Scaling

The data from the selected features is needed to be normalized before applying training and testing. The purpose of applying normalization is to bring all the numerical features in the same scale without disturbing the range differences. The normalized scale for each feature is obtained using Equation 3.1, where FV is the current feature value, min(FV) is the minimum feature value in the current feature and max(FV) is the maximum feature value in the current feature.

$$NS = \frac{FV - min\left(FV\right)}{max\left(FV\right) - min\left(FV\right)} \tag{3.1}$$

### 3.1.6 Summary

Testing different machine learning classifiers in NTL detection requires a real-world dataset. This chapter introduced our dataset with the explanation of each feature. The pre-processing steps require the data to be transformed into a form suitable for down stream analytical processing. This chapter also explained each transformation step in detail. Lastly, it also highlighted the feature selection, data merging and feature scaling process.

# Chapter 4

# Performance Analysis of Machine Learning Classifiers for Non-Technical Loss Detection

## 4.1 Introduction

The process of on-site inspection for a potential detection of NTL incurs a heavy cost to the company and it is practically impossible to inspect a large number of households. This can only be fruitful if the inspection results in a large number of NTL detection. In reality, the ratio of number of NTL detection to the number of inspections is generally very low for companies (Coma-Puig et al., 2016). This can also be explained due to the fact that people may have got their second homes, or they may be on long vacations, etc. Shortlisting them for the inspection will only increase the inspection cost.

Over the past decade, the research community has paid attention to detecting NTL with the collaboration of electric suppliers using machine learning classifiers. This includes using Support Vector Machine (SVM), Optimum Path Forest (OPF), random forest, multi-layer perceptron neural network (NN), *K*-Nearest Neighbors (KNN), Adaboost, naive bayes, decision trees and deep learning. Training sets containing records of NTLs are used to train the models and test sets are used to evaluate them. The list of fraudsters identified by the classifiers is then used for on-site inspection. The hit ratio of NTL detection using machine learning classifiers is very promising as compared to random guessing of potential fraudsters. To compare the performance of these classifiers, different performance evaluation metrics are used. These metrics can help in shortlisting the classifiers for NTL detection under given scenarios.

A detailed comparative study of performance evaluation metrics is still needed to diagnose the relationship between different metrics when used for NTL detection. These relationships have not been discussed sufficiently in the literature of NTL and can be proved to be a baseline for the selection of appropriate performance evaluation metrics for NTL detection. In Pakistan, electric supply companies perform random on-site inspection to identify theft cases. This is the prime reason for a very small success on NTL detection. Our contribution in this regard helps to improve the hit ratio in an electric supply company in Pakistan by identifying and shortlisting the potential theft cases for on-site inspection using machine learning classifiers. This work is performed on a real dataset which is thoroughly explained in Chapter 3. It will also help to reduce the on-site inspection cost of the company. Due to the success of random forest, KNN and linear SVM in the class imbalance domain, this work first uses these three classifiers to predict the occurrences of NTL in the dataset. Then, it computes 14 performance evaluation metrics across the three classifiers to identify the key scientific relationships between these performance metrics with respect to NTL detection. For the appropriate selection of the classifiers, these relationships are crucial. Therefore, we have focused on identifying the key scientific relationships between performance evaluation metrics in the domain of NTL detection as shown in Figure 4.1. This work can further be extended to predict potential theft in the gas sector. Using the consumption pattern of gas consumers, this set of classifiers along with the performance metrics can be used for the identification of gas theft attempts.

The objectives of this chapter are as follows:

1. Categorize the state of the art NTL detection schemes and present their comprehensive taxonomy.

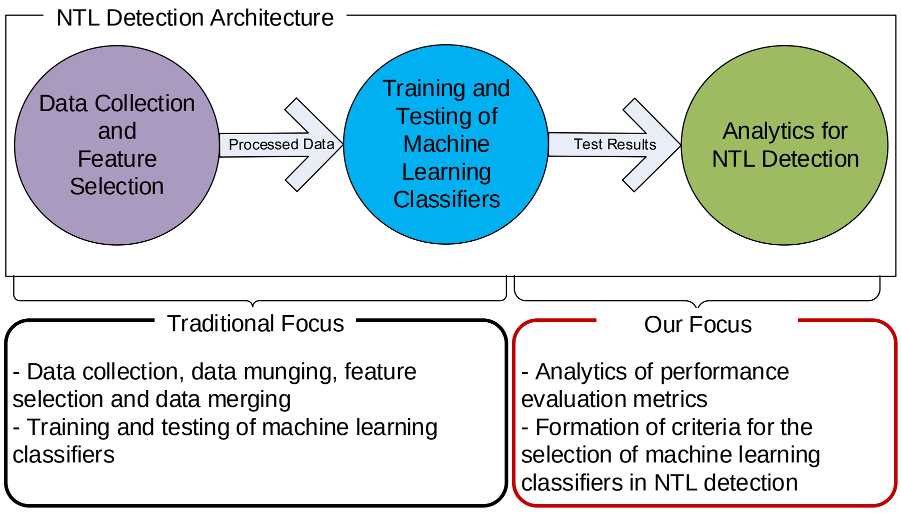
NTL Detection Architecture
Data Collection and Feature Selection
Processed Data
Training and Testing of Machine Learning Classifiers
Test Results
Analytics for NTL Detection
Traditional Focus
- Data collection, data munging, feature selection and data merging
- Training and testing of machine learning classifiers
Our Focus
- Analytics of performance evaluation metrics
- Formation of criteria for the selection of machine learning classifiers in NTL detection

FIGURE 4.1: The NTL Architecture comprising of data collection, feature selection, training and testing of machine learning classifiers and analytics. Note that existing research works mostly focus on NTL detection while analytics of performance evaluation metrics have often been overlooked.

2. Identify the strengths and weaknesses of the state of the art methods for NTL detection and identify a pool of performance metrics commonly used for NTL detection.

3. Apply machine learning classifiers for NTL detection, and compute and validate their performances on a real dataset containing approximately 80, 000 monthly records of electricity consumption.

4. Investigate a pool of the identified performance metrics for NTL detection and highlight those metrics that can best describe the identification of NTL.

## 4.2 Methodology

In this section, we first describe the proposed methodology used for NTL detection in the electric distribution company. Then, we outline the need of separate performance evaluation metrics for NTL detection. Finally, we discuss a number of such existing metrics which proved to be good for NTL detection. The proposed methodology consists of seven steps which are described in the following subsections. The first step is data collection from the company. The data contains monthly consumption records of electric consumers. The data is collected in a comma separated values (CSV) file which needs to be converted into a form suitable for analytical processing. Data munging performs this functionality along with steps like duplicate removal and dealing with NULL values, etc. Not all features are useful for the analytic process. Feature selection step shortlists the features which are most useful in predicting NTL. The company separately maintains the data of risky consumers. The data merging step combines the selected features with the data of risky consumers. Once the features and the records are finalized, the next step is to normalize all features. This is done by the scaling step. Next, training and testing of the classifiers is performed. On the basis of the results obtained from testing, different performance evaluation metrics are then calculated which form a strong foundation in identifying different criterion for the selection of suitable classifiers for NTL detection. The complete methodology is shown in Figure 4.2, while Figure 3.1 shows the pre-processing steps only.

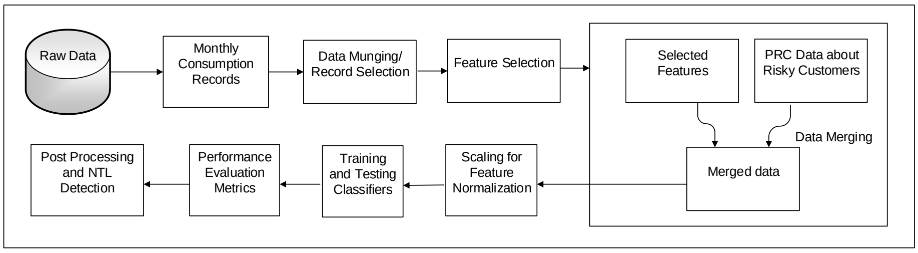
Raw Data
Monthly Consumption Records
Data Munging/ Record Selection
Feature Selection
Selected Features
PRC Data about Risky Customers
Data Merging
Merged data
Scaling for Feature Normalization
Training and Testing Classifiers
Performance Evaluation Metrics
Post Processing and NTL Detection

FIGURE 4.2: The proposed methodology for NTL detection

### 4.2.1 Data Collection and Data Munging

NTL detection cannot be thoroughly studied without a real dataset. We have collected a dataset from an electric supply company in Pakistan. The details of the collected data and the data munging step are thoroughly discussed in Sections 3.1.1 and 3.1.2 in Chapter 3, respectively.

### 4.2.2 Feature Selection

From the set of 71 features, a subset of 14 useful features is shortlisted using feature importance. It is a measure of finding the importance of each feature (Breiman, 2001). A feature has an importance if the model's error of prediction is increased with the shuffling of the value of the feature. The increase in model's prediction error indicates that the model relies on that feature. Thus, the feature is important. Conversely, a feature has less importance if the model's error of prediction is not changed with the shuffling of the value of the feature. The stability in the model's prediction error indicates that the model does not rely on that feature. Thus, the feature is not important. To obtain the list of useful features, the 71 features are first listed in descending order with respect to feature importance. Then, using the Gini Index, a threshold for the optimum number of features is selected beyond which including any other feature should not affect the F-measure. This way, we have not only found the optimum combination of features for which the F-measure is best but it also has significantly reduced the computational time of the classifiers. The list of shortlisted features, their description and the feature importance is enlisted in Table 4.1.

### 4.2.3 Data Merging and Scaling

The details of data merging and scaling steps are discussed in Sections 3.1.4 and 3.1.5 in Chapter 3, respectively.

TABLE 4.1: Feature Description and Feature Importance of Selected Features, Sorted by their Importances in Descending Order

| Feature | Description | Importances |
|---|---|---|
| Units-12Months | Units consumed during last 12 months | 0.141807732008 |
| Amount-12Months | Total amount billed during last 12 months | 0.116774856878 |
| BilledUnit-YTD | Units billed in current year | 0.116751907486 |
| BilledAmount | Amount billed in current month | 0.092791430138 |
| 1 year LPS | Late payment surcharge in last one year | 0.057126632724 |
| Amount-12MonthsAvg | Average monthly amount in last 12 months | 0.040861765397 |
| BilledAmount-12MonthsGross | Total payment made in last 12 months | 0.038509950395 |
| Units-12MonthsAvg | Average monthly units in last 12 months | 0.032045710111 |
| Amount-GrossBilledYTD | Total payment made in current year till date | 0.029088274439 |
| Amount-12MonthsAvgGrossBilled | Total average monthly payment made in last 12 months | 0.028248805369 |
| Amount-Regular | Payable amount for regular units | 0.023397470434 |
| 1 Month LPS | Late payment surchare in last 30 days | 0.022879096842 |
| Month-Billing | Month of billing | 0.016062708402 |
| InstallementNo | Number of installements | 0.015831709285 |

### 4.2.4 Training and Testing

The normalized data obtained from scaling steps is used for training and testing of the three classifiers namely KNN, random forest and SVM. The theoretical details of these classifiers can be found in Section 2.2.2.

### 4.2.5 Post-Processing and NTL Detection

For the last few years the research community has been paying attention on deriving methods which focus on representing the evaluation of classes separately. Table 4.2 shows the basic confusion matrix, which is used to formulate more complex metrics for datasets containing imbalanced class distribution. For NTL, True Positive (TP) is the instances of theft cases correctly classified by the classifier and True Negative (TN) is the instances of normal cases correctly classified by the classifier. False Positive (FP) indicates instances of normal cases identified as theft by the classifier and False Negative (FN) indicates the instances of theft cases identified as normal by the classifier. The metrics are then used to calculate more complex metrics like accuracy, recall, precision, TNR, FPR, FNR, NPV, $F_\beta$, arithmetic mean, harmonic mean, G-Mean and dominance. These metrics are discussed in Section 4.2.6. These metrics yield a set of

TABLE 4.2: Confusion Matrix

| | Predicted Positive | Predicted Negative |
|---|---|---|
| Actual Positive | True Positive (TP) | False Negative (FN) |
| Actual Negative | False Positive (FP) | True Negative (TN) |

comprehensive observations particularly related with NTL detection. The observations are discussed in Section 4.3.

### 4.2.6 Evaluation Metrics

Datasets from electric industry have a strong imbalanced distribution of target variable. Doing predictive modeling in these datasets is a challenging task due to the fact that distribution of classes (target variable Y) is non-uniform. It could be a case that training and testing samples contain 99% of total samples belonging to the normal class and the remaining 1% belong to the thief class. The scenario becomes more complex when the user's choice is biased towards the least represented class, i.e. the thief class. Performance metrics that are used for the balanced datasets can not be efficiently used for datasets with imbalance distribution of target variable as these metrics tend to ignore the thief class for which the performance measure is actually needed. Thus, giving the performance measures against the unwanted and the most repeated class is not helpful in accessing the performance of the least represented class predictions. Therefore, accuracy and error rate are not the right measures as they are biased towards the normal class (Manning, Raghavan, and Schütze, 2008). In fact, we need measures which evaluate the correctness of the normal and the thief class separately. For this, a basic confusion metric is used to calculate True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). The basic confusion metric is further used to evaluate more complex metrics including precision, recall, arithmetic mean, harmonic mean, NPV, $F_\beta$, G-mean, dominance, TPR, TNR, FPR and FNR. The details of these metrics are presented in Section 2.2.3.

## 4.3 Results and Analysis

In this section, we first perform extensive simulation of the random forest, KNN and SVM on training and test data. The three classifiers are chosen due to their success in the imbalance datasets. The experiments are performed using Python's open source library, scikit-learn (Pedregosa et al., 2011b). Then, we discuss a detailed analysis of the comparison of performance evaluation metrics across the three classifiers along with the comparison of the classifiers. A list of simulation parameters is also presented in Table 4.3

TABLE 4.3: List of Simulation Parameters

| Classifier | Simulation Parameters |
|---|---|
| Linear SVC | penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, verbose=0, random_state=None, max_iter=1000 |
| KNN | n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None |
| Random Forest | n_estimators='warn', criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None |

### 4.3.1 Experimental Setup

The experiments were performed on a 64-bit Windows server with Intel Xeon 2.2 GHz processor and 32 GB RAM. All the algorithms were implemented in Python 3.6. The total number of transaction records is $80,244$ out of which $64,195$ are selected for training the three classifiers namely random forest, KNN and SVM. The remaining $16,049$ records are selected for testing the classifiers. The percentage representation of the training set is 80% while for the test set, it is 20%. The training time for random forest, KNN and SVM is recorded as 22 seconds, 2 seconds and 30 seconds, respectively. The training time is represented in Table 4.4.

The values of TP, TN, FP and FN for the test set across the three classifiers are listed in Table 4.5. The values of the other complex performance metrics for the three classifiers are listed in Table 4.6. The performance of the classifiers can vary with the change of dataset as

TABLE 4.4: Training Time of Random Forest, KNN and SVM

| Classifier | Training Time |
|---|---|
| Random Forest | 22 Seconds |
| *K*-Nearest Neighbors | 2 Seconds |
| Support Vector Machine | 30 Seconds |

it depends on the selected features. A different dataset with a different set of features can result in increase or even decrease of performance. So, the better the feature set, the higher the performance will be.

TABLE 4.5: TP, TN, FP and FN values across random forest, KNN and SVM

| | Random Forest | *K*-Nearest Neighbors | Support Vector Machine |
|---|---|---|---|
| TP | 677 | 678 | 672 |
| TN | 15,347 | 15,343 | 15,338 |
| FP | 19 | 23 | 28 |
| FN | 6 | 5 | 11 |
| Total | 16,049 | 16,049 | 16,049 |

KNN outperformed random forest and SVM in terms of TP. It has the maximum instances of theft detection which is 678. For random forest, TP is 677 and for SVM, it is 672. Accuracy for the three classifiers are approximated to 99% but seeing precision, it is observed that random forest performed better than KNN and SVM, while KNN outperformed random forest and SVM on the basis of recall as it has the best recall of 99.27%. Random forest has the highest arithmetic mean and harmonic mean.

### 4.3.2 Comparison of precision and recall

An important observation regarding the problem of NTL detection is that the model which has a high recall is most suitable for theft detection. In order to understand this relation, consider the cases of FP and FN. False Positives are those normal users that have been predicted by the classifier as thieves whereas False Negatives are those thieves that are predicted by the classifier as normal users. Considering the two cases, having a large FP value will only result in

TABLE 4.6: Other complex metrics for the three classifiers

| | Random Forest | $K$-nearest Neighbors | Support Vector Machine |
|---|---|---|---|
| Accuracy (%) | 99.84% | 99.83% | 99.76% |
| Precision (%) | 97.27% | 96.71% | 96.0% |
| Recall (%) | 99.12% | 99.27% | 98.39% |
| Arithmetic Mean (%) | 98.20% | 98.0% | 97.19% |
| Harmonic Mean (%) | 98.19% | 97.98% | 97.18% |
| NPV | 1.0 | 1.0 | 0.999 |
| $F_\beta$ (for $\beta = 1$) | 98.2 | 98.0 | 97.2 |
| $F_\beta$ (for $\beta = 1.5$) | 98.5 | 98.5 | 97.6 |
| G-Mean | 99.50 | 99.56 | 99.10 |
| Dominance | -0.008 | -0.006 | -0.014 |
| TPR | 0.991 | 0.993 | 0.984 |
| TNR | 0.999 | 0.999 | 0.998 |
| FPR | 0.001 | 0.001 | 0.002 |
| FNR | 0.009 | 0.007 | 0.016 |

increasing the manual effort of on-site inspections whereas a high FN value will result in the failure of the classifier to correctly identify the thieves. Therefore, for NTL it is recommended to promote the classifier which has a low FN value. Now, considering the Equation 2.15, it can be observed that recall increases with the decrease of FN. This gives a nice measure of the selection of the classifier for NTL detection that both the precision and the recall should not have equal priority. In fact, for NTL detection, classifiers with high recall are most suitable regardless of what the precision value is. In Table 4.5, it is observed that KNN has the lowest number of FN, i.e. 5. Consequently, it has the highest recall among the three classifiers as shown in Figure 4.3. The lowest recall is observed for SVM, which is 98.39%. Thus, the percent increase of recall from using SVM to KNN is 0.89%. This gives a clear indication that for our real dataset, KNN is the better choice for NTL detection. For two classifiers having the same recall but different precision values, the classifier with a high precision should be selected. Precision will increase with the decrease in FP. So, when the two classifiers have equal recalls, the classifier having the lowest FP should be given preference. This observation can be verified by Equation 2.19. For all the three classifiers, recall is observed higher than their corresponding precision values. It is further observed that SVM has the lowest precision and recall among the three classifiers as shown in Figure 4.3.

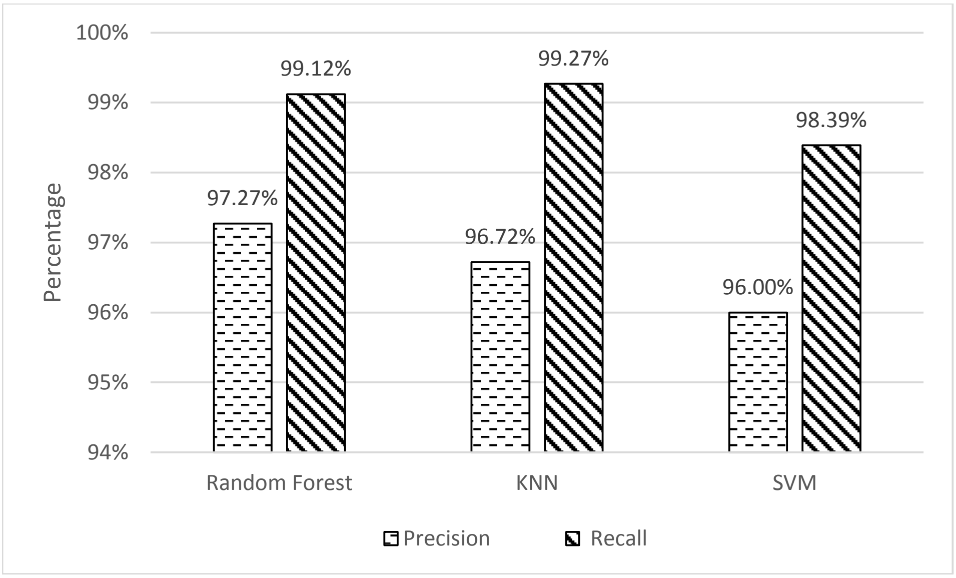

FIGURE 4.3: Comparison of precision and recall

### 4.3.3 Comparison of accuracy, FPR and FNR

Total normal users that are predicted as thieves is measured by FPR. A high FPR increases the on-site inspection for theft verification, which consequently results in increase of manual efforts. On the other hand, a high FPR also indicates the success of the classifier in identifying thieves that are categorized as normal users in the company. Another measure that gives a close insight of the number of thieves that are wrongly classified as normal users is FNR. A low FNR is desirable in NTL detection. Seeing the accuracy measure, all the three classifiers looked to be performing exceptionally well but observing FPR and FNR, it is found that accuracy is not depicting the facts about FP and FN. Figure 4.4 shows that FPR is very low for the three classifiers. For KNN, FNR is the least among the three classifiers showing that it has the lowest FN value and thus, KNN is found to be a good choice for NTL in our real dataset. Among the three classifiers, the highest FNR is observed for SVM showing that it has the highest value of FN which can also be verified in Table 4.5. Thus, for our real dataset, SVM turns out to be the last choice for NTL detection.

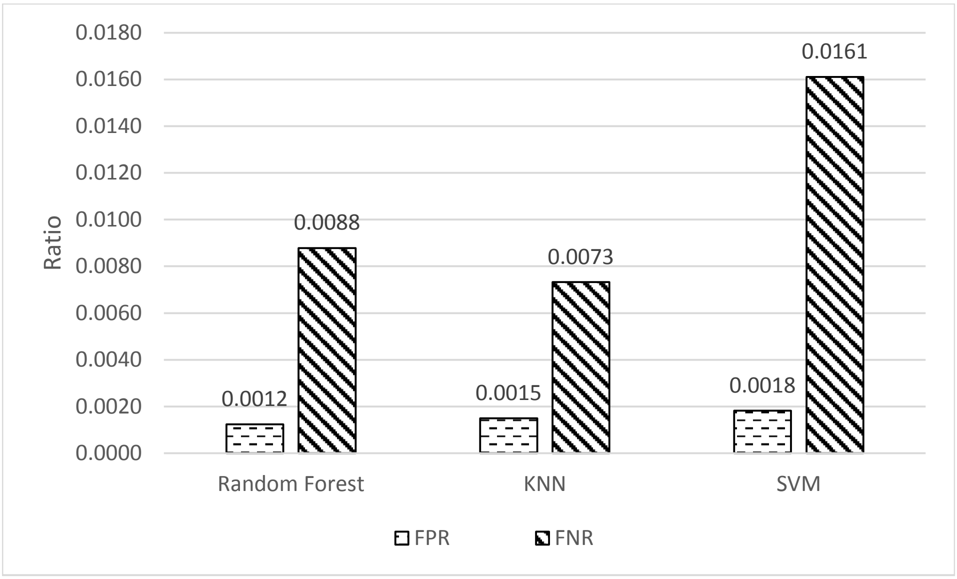

FIGURE 4.4: Comparison of FPR and FNR

### 4.3.4 Comparison of F-Measures

$F_{\beta=1}$ and $F_{\beta=1.5}$ are close to each other in all classifier readings. Both are high for the three classifiers depicting that both precision and recall values for the classifiers are considerably high. The lowest reading for the two metrics are observed for SVM showing that precision and recall values for SVM are lower as compared to their counterparts, which can also be verified using Table 4.6. Thus, SVM is the last choice for NTL detection in this real dataset. For $F_{\beta=1}$, random forest has a higher value than KNN and for $F_{\beta=1.5}$, random forest and KNN have equal values. Considering $F_{\beta=1}$, random forest has performed better than KNN and considering $F_{\beta=1.5}$, both random forest and KNN have equal performance. Given that recall has a high weightage in $F_{\beta=1.5}$, for all the classifiers, $F_{\beta=1.5}$ is high as compared to the corresponding $F_{\beta=1}$. This indicates that recall is high for all the classifiers as compared to precision. The percentage increase from precision to recall in random forest, KNN and SVM is 1.9%, 2.65%, and 2.49%, respectively. The highest increase in percentage is observed for KNN and thus, it also has the highest difference of values between $F_{\beta=1}$ and $F_{\beta=1.5}$, i.e. 0.5. This indicates that KNN outperformed random forest and SVM. Also, $F_\beta$ values are observed to be between recall and precision values for all classifiers as shown in Figure 4.5. This shows that $F_\beta$ of precision and recall behaves just like the harmonic mean. As discussed in Section 4.3.2, for NTL detection recall should be given high priority as compared to precision. This can be achieved by using $F_\beta$ measure with $\beta$ value greater than 1.

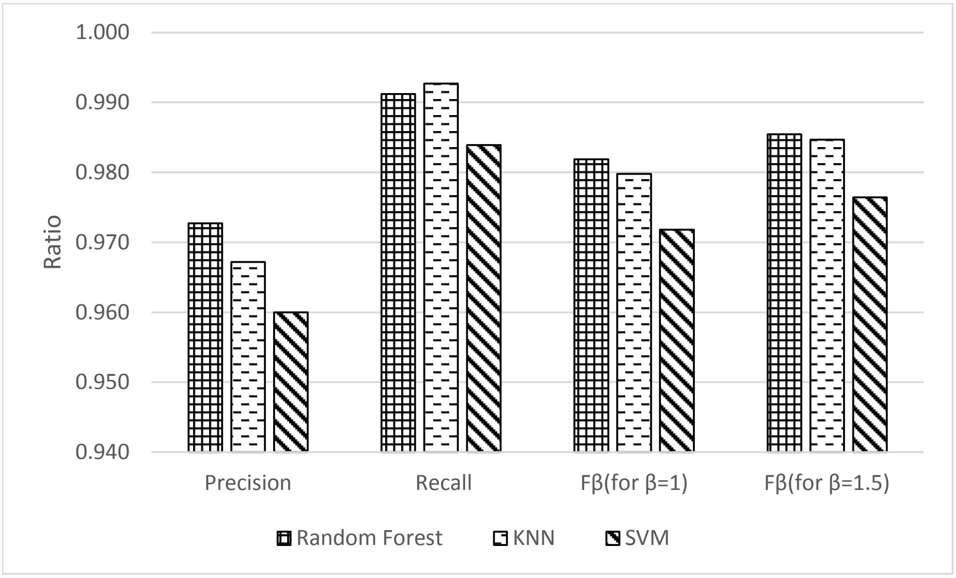

FIGURE 4.5: Comparison of precision, recall, $F_{\beta=1}$ and $F_{\beta=1.5}$

## 4.3.5 Harmonic mean

For all the classifiers, harmonic mean is lower than the arithmetic mean. Harmonic mean is also observed to be closer to the smaller of the precision and recall for all classifiers. Random forest has the highest harmonic mean. This indicates that not only the precision and recall values for random forest are high but also they are close to each other. This can be verified by the fact that random forest has the smallest percentage increase from precision to recall, i.e. 1.9%.

Harmonic mean for SVM is lowest among the three classifiers showing that the corresponding values of precision and recall for SVM are also low, as shown in Figure 4.6. Therefore, instead of maintaining both the precision and the recall, harmonic mean can also be used for the evaluation of the classifiers in NTL detection.

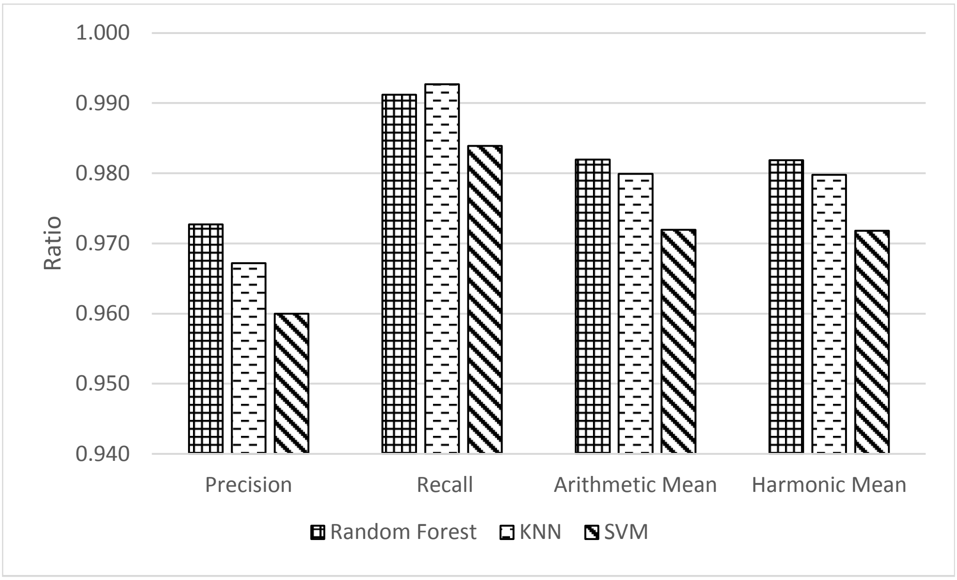

FIGURE 4.6: Comparison of precision, recall, arithmetic mean and harmonic mean

### 4.3.6 Comparison of TPR, TNR and G-Mean

For all the classifiers, G-Mean is high. This indicates that TPR and TNR for the three classifiers are also high. G-Mean for SVM is lowest among the three classifiers indicating that its TPR is also lowest as shown in Figure 4.7. Therefore, it can be deduced that for NTL detection, a classifier with a high G-Mean value is preferable over a classifier with a low G-Mean value. Thus, KNN outperformed random forest and SVM.

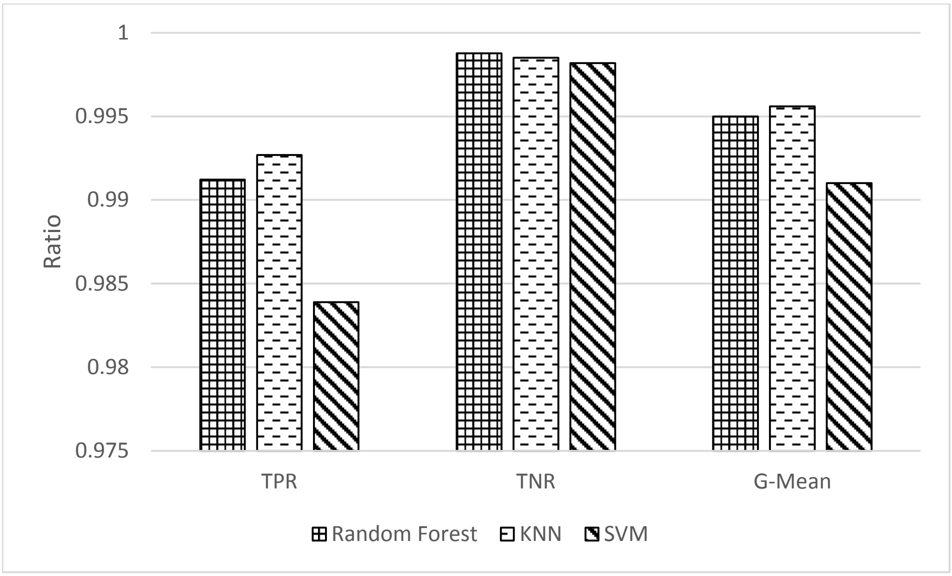

FIGURE 4.7: Comparison TPR, TNR and G-Mean

## 4.3.7 Comparison of TPR, TNR and dominance

A classifier having dominance close to $-1$ depicts that it has a high TNR but a low TPR. In contrast, a classifier having dominance close to 0 indicates that it is good in predicting both classes for NTL detection. For NTL detection, TPR and TNR give close insight of the performance of a classifier. Combining TPR and TNR, dominance gives a good choice of a performance evaluation metric for NTL detection. For our dataset, comparison of TPR, TNR and dominance is shown in Figure 4.8. It is observed that among the three classifiers, KNN has the dominance closest to 0. This shows that for our dataset, KNN is the best in predicting both classes.

## 4.3.8 Comparison of NPV and FNR

For NTL detection, occurrence of theft instances is rare while normal consumers are in huge number. As NPV indicates the number of normal consumers only and ignores the theft cases, therefore for NTL detection, NPV is not a suitable metric. It is observed that for

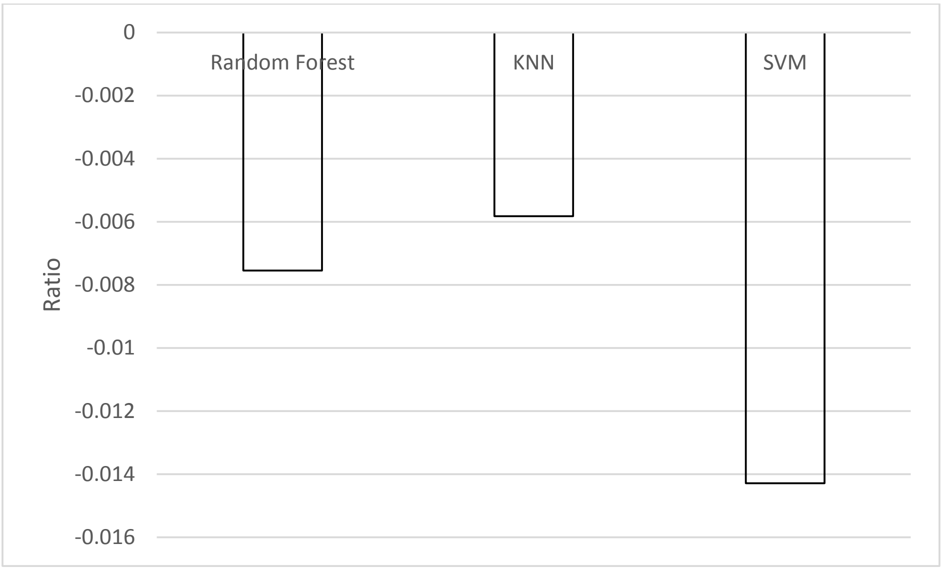

FIGURE 4.8: Dominance of Random Forest, KNN and SVM

all the classifiers, NPV is close to 100%. A clear reason for this is that NPV is ignoring theft cases and considering normal consumers only. In contrast, FNR is a measure of number of thieves that are predicted as normal consumers. For NTL detection, this ratio is needed to be as low as possible. It is observed that KNN has the lowest FNR. Thus, KNN is a good choice for NTL detection. For the three classifiers, FNR is shown in Figure 4.9. The figure shows that SVM has the highest FNR and thus, for our dataset, it is the last option for NTL detection.

## 4.4 Conclusion and Future Work

This work has used a real-world dataset of an electric supply company in Pakistan to identify the non-technical loss by applying three classifiers namely random forest, *K*-nearest neighbors and linear support vector machine. The classifiers are chosen due to their success

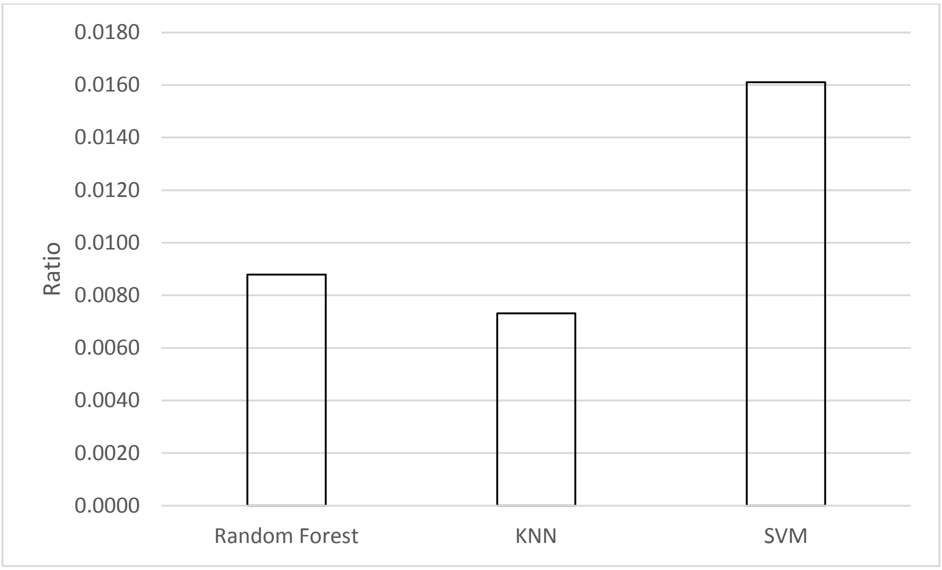

FIGURE 4.9: FNR of Random Forest, KNN and SVM

in class imbalance domain. The aim of the study is to use these classifiers to first identify existing NTL attempts and then predict new theft cases.

It further uses 14 different metrics to perform an in-depth performance analysis of the three classifiers. One of the core findings of this contribution is that for NTL detection, both the precision and recall should not have equal precedence. In fact, the classifier with a higher recall is better. The percent increase of recall from using KNN to random forest is 1.24%. This clearly depicts that random forest is the better choice for NTL detection as it has the higher recall. This analysis can be used as a baseline for the accurate selection of the classifiers in NTL detection. This work will vastly benefit the electric supplier in detecting NTL. It will not only improve their abilities for NTL detection, but will also save huge amount of monetary losses which they are already bearing.

There is a need to further extend the use of performance evaluation metrics that can estimate and compare error rates on the basis of which a combination of classifiers can be selected for a specified dataset for NTL detection. Currently, there is a small range of graphical metrics used for performance analysis. This includes receiver operating characteristic (ROC) and area under ROC curve (AUC) ("A Survey of Predictive Modeling on Imbalanced Domains"). There is also a need of further exploration in the usage of graphical performance metrics. Furthermore, the performance of classifiers with respect to their categories is another future direction for NTL detection.

# Chapter 5

# Analyzing Types of ML Classifiers for NTL Detection

## 5.1 Introduction

NTL detection involves training different machine learning classifiers with existing data that contains observations from both positive and negative classes. After training, the classifiers are then tested on a different set of test data. The test results are then evaluated using performance evaluation metrics. The use of machine learning classifiers for NTL detection has been an ongoing and interesting activity in the research community that has now span for over two decades. There are many types of the classifiers that have been tested for NTL detection. These include decision trees, ensemble methods, generalized linear models, linear and quadratic discriminant analysis, Naive Bayes, nearest neighbors, neural network models, stochastic gradient descent and support vector machines.

Machine learning classifiers are by far the most flexible way of NTL detection for many reasons. Unlike manual on-site inspection, machine learning requires some technically skilled professionals who can work on real datasets to identify the occurrence of theft. The shortlisted theft cases can then be verified by manual inspection. Adding new theft cases to the learning models makes the models learn new cases. In this way, the performance of the classifiers keeps improving. These machine learning methods are also useful to identify real culprits unlike other network-based techniques which are only able to identify an area where NTL is committed but fail to pinpoint the theft cases. Furthermore, a much less cost is incurred in using machine learning methods as compared to other procedures like manual on-site inspection. Automation of the NTL detection procedure is another added advantage of using machine learning. This advantage can not be gained while using other NTL detection schemes like manual on-site inspection.

The main problem is that a detailed comparative study of the machine learning classifiers on some real dataset is still missing. This work contributes in identifying NTL in a real dataset taken from a power supply company in Pakistan. A comprehensive explanation of the dataset is presented in Chapter 3. NTL in power sector is given less importance in Pakistan and less effort is done for its detection. That is why an on-site inspection is the only measure deployed for the detection of NTL. In this chapter, our objectives are as follows:

1. Present a taxonomy of the NTL detection techniques and categorize the strategies of NTL detection with respect to data, network, a combination of both and additional data. Moreover, identify and focus on a number of solutions based on unsupervised, supervised, semi-supervised and hybrid learning.

2. For NTL detection, use a real dataset taken from a power supply company in Pakistan. The contributions which use synthesized data for NTL detection generally contain equal distribution of classes that do not depict the natural class distribution. As our dataset is real, the ratio between positive and negative classes is imbalanced which represents the natural behavior.

3. Investigate the performance of different types of machine learning classifiers which are based on the algorithms they use and identify the type that perform best in NTL detection. The types involve ensemble methods, Neural Networks (NN), Decision Trees (DT), Nearest Neighbors, Stochastic Gradient Descent (SGD), Generalized Linear Models, Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA), Support Vector Machine (SVM) and Naive Bayes. Moreover, identify a threshold of features beyond which adding more features does not affect the efficiency of the classifiers.

4. Perform extensive simulations and find the classifier that is dominating in terms of F-measure and Recall as compared to

counterparts. It will open a potentially new area where NTL detection can be worked on.

## 5.2 Methodology

This section outlines the data collection, feature selection strategy and the metrics used to evaluate the performance of the classifiers.

### 5.2.1 Data Collection

For NTL detection, a real dataset is collected from a power supply company in Pakistan. The details of the collected data and the data munging step are thoroughly discussed in Sections 3.1.1 and 3.1.2 in Chapter 3, respectively.

### 5.2.2 Pre-Processing

Initially, a set of 71 features is selected that span across six major categories as illustrated in Appendix A. These include normal amount, normal units, additional amount, additional units, bill info and extra info. A detailed explanation of each feature is described in Chapter 3.

### 5.2.3 Selecting Top-*k* Features

One of the contributions of this paper is to find the optimum number of *k* features that can provide best theft prediction in a real dataset. It is observed that not every feature has an equal or comparable participation in predicting NTL. Some features have a high role while others have a negligible role. Also, using all 71 features to predict NTL will increase the computational complexity of the classifiers. It turns out that there should be a threshold for the contributing features beyond which including or excluding features should not affect the efficiency of the classifier. For this, we first sort the feature set in descending order with respect to feature importance (Liaw, Wiener, et al., 2002b). It is a measure that uses accuracy to filter attributes which are most suitable for correctly identifying the target variable.

Thus, it gives an insight to the relative importance of every feature with respect to the target variable. For a theoretical definition of feature importance, the reader can refer to (Breiman, 2001). Then, we apply Gini Index to find the top-$k$ number of features for which the F-measure is the highest where $k$ ranges from 1 to 71. As described in (Han, Kamber, and Pei, 2011), the Gini Index is used to quantify the impurity of data partition D. The mathematical representation of Gini Index is given in Equation 5.1.

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2 \tag{5.1}$$

where $p_i$ is the probability estimate of a record belonging to the class $C_i$ out of a total of $m$ classes. Finally, the $k^{th}$ value for which the best F-measure is found is selected. For our dataset, the value of $k$ with best F-measure is 14. This indicates that using this set of top 14 features to find NTL has the same behavior as using all 71 features. This simulation has not only identified key features that are participating in predicting NTL in the real dataset but it also has helped to significantly reduce the execution time of the classifiers. Table A.1 lists 71 features and their corresponding feature importance. The cumulative percentage of feature importance of top 14 features is presented in Figure 5.1, which shows that the contribution of top 14 features in predicting NTL is 77%.

### 5.2.4 NTL and Evaluation Metric

NTL detection is an application of imbalance problem domain. It is a problem where the dataset is highly biased towards one of the outcomes of the target variable while the other outcome(s) remains least representative. Interestingly, the focus is on the least representative outcome. This leads to the requirement of an appropriate selection of the evaluation metric. Taking the example of NTL detection, most of the users are not thieves (True Negative), while few are thieves (True Positive). Now, selecting accuracy as an evaluation metric would be a wrong choice as the results will be highly biased towards the most representative class, i.e. TN. In fact, we

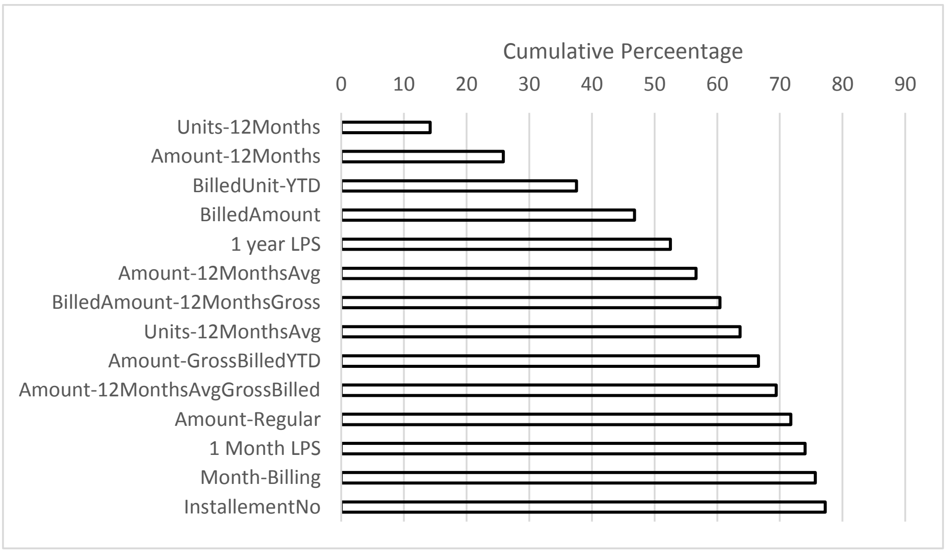

FIGURE 5.1: Cumulative Percentage of Feature Importance of Top 14 Features

need a measure which should comprehensively give an insight to the actual number of thieves (recall) as well as the actual number of predicted thieves (precision) along with the combination of the two. For this, F-measure is used, which combines precision and recall. In this work, we have used precision, recall and F-measure as our performance evaluation metrics. The details of these metrics are presented in Section 2.2.3.

### 5.2.5 Models

A variety of different machine learning algorithms are tested for NTL detection which also include recently developed ensemble methods namely CatBoost, LGBoost and XGBoost. We investigated the types of classifiers mentioned in Figure 5.2 to find the best classifiers for NTL detection. The figure represents the classifiers and their types that are used in our experiments. The description of each classifier and its type is presented in Section 2.2.2.

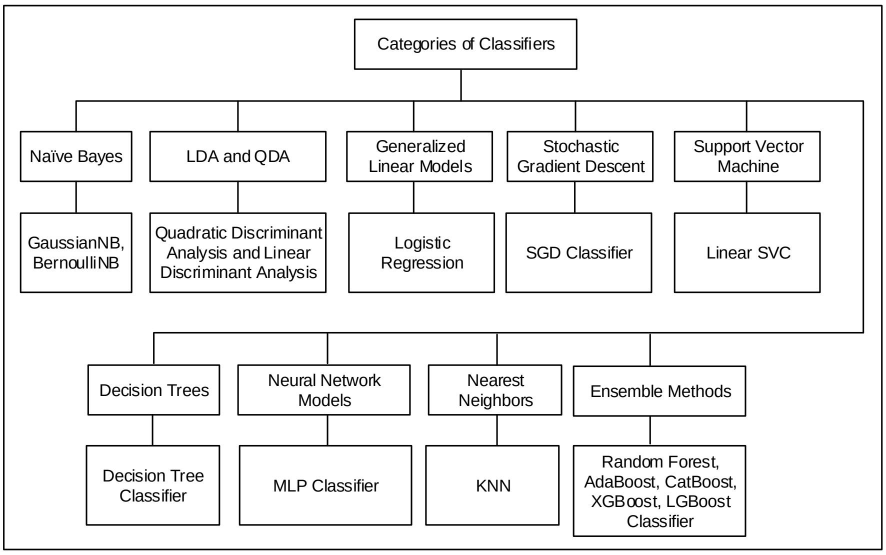

FIGURE 5.2: Hierarchy of Classifiers

## 5.3 Experimental Results

In this section, we validate our work by performing extensive simulations using Python 3.6 on a 64-bit Windows Server with hardware specification including an Intel Xeon 2.2 GHz processor and 32 GB RAM. Other than CatBoost, LGBoost and XGBoost, all the classifiers are trained and tested using the scikit-learn (Pedregosa et al., 2011b) open source library for Python. CatBoost [1] (Prokhorenkova et al., 2018), LGBoost [2] (Ke et al., 2017) and XGBoost [3] (Chen and Guestrin, 2016) are also open source libraries which are available on GitHub for Python. Then, we perform a detailed analysis of the results of the classifiers which span across 9 different types. A list of simulation parameters of the classifiers is presented in Appendix B.

[1]https://github.com/catboost/catboost
[2]https://github.com/microsoft/LightGBM
[3]https://github.com/dmlc/xgboost/tree/master/python-package

### 5.3.1 Performance Analysis of Various Types of Classifiers on Reduced Feature-Set

The top-14 features identified in feature selection process are used to calculate the confusion matrix, precision, recall and F-measure of different machine learning classifiers. The confusion matrix for all classifiers is presented in Table 5.1.

TABLE 5.1: TP, TN, FP and FN of all classifiers

| Type of Classifier | Classifiers | TP | TN | FP | FN |
|---|---|---|---|---|---|
| Naive Bayes | BernoulliNB | 1 | 15365 | 1 | 682 |
| | GaussianNB | 58 | 15249 | 117 | 625 |
| LDA and QDA | LDA | 230 | 15302 | 64 | 453 |
| | QDA | 566 | 15167 | 199 | 117 |
| Generalized Linear Models | LogisticRegression | 453 | 15351 | 15 | 230 |
| SGD | SGDClassifier | 639 | 15336 | 30 | 44 |
| Support Vector Machine | LinearSVC | 672 | 15338 | 28 | 11 |
| Decision Tree | DecisionTreeClassifier | 672 | 15346 | 20 | 11 |
| Neural Network Models | MLPClassifier | 679 | 15341 | 25 | 4 |
| Nearest Neighbors | KNeighborsClassifier | 678 | 15343 | 23 | 5 |
| Ensemble Methods | AdaBoostClassifier | 660 | 15350 | 16 | 23 |
| | RandomForestClassifier | 677 | 15347 | 19 | 6 |
| | LGBoost | 606 | 15356 | 10 | 77 |
| | XGBClassifier | 674 | 15349 | 17 | 9 |
| | CatBoostClassifier | 677 | 15352 | 14 | 6 |

One of our contributions is that we compare the results of the classifiers with respect to the types of the classifiers. Another contribution is that we compare the efficiency of the recently developed CatBoost, LightGBM and XGBoost with other supervised machine learning classifiers including Gaussian Naive Bayes, Bernoulli Naive Bayes, Quadratic Discriminant Analysis (QDA), Stochastic Gradient Descent (SGD), Decision Trees, Random Forest, K-Nearest Neighbors (KNN), Adaboost, Multi-Layer Perceptron (MLP) classifier, Linear SVC, Logistic regression and Linear Discriminant Analysis (LDA). The precision, recall and F-measure for all the classifiers are presented in Table 5.2.

#### 5.3.1.1 Performance Analysis of Classifiers with Respect to their Types

It is interesting to observe that considering F-measure as the classifiers' efficiency measure, ensemble methods outperformed all other

TABLE 5.2: Precision, Recall and F-measure of all classifiers

| Type of Classifier | Classifiers | Pre. | Recall | F-M |
|---|---|---|---|---|
| Naive Bayes | BernoulliNB | 0.500 | 0.001 | 0.003 |
| | GaussianNB | 0.331 | 0.085 | 0.135 |
| LDA and QDA | LDA | 0.782 | 0.337 | 0.471 |
| | QDA | 0.740 | 0.829 | 0.782 |
| Generalized Linear Models | LogisticRegression | 0.968 | 0.663 | 0.787 |
| SGD | SGDClassifier | 0.955 | 0.936 | 0.945 |
| Support Vector Machine | LinearSVC | 0.960 | 0.984 | 0.972 |
| Decision Tree | DecisionTreeClassifier | 0.971 | 0.984 | 0.977 |
| Neural Network Models | MLPClassifier | 0.964 | 0.994 | 0.979 |
| Nearest Neighbors | KNeighborsClassifier | 0.967 | 0.993 | 0.980 |
| Ensemble Methods | AdaBoostClassifier | 0.976 | 0.966 | 0.971 |
| | RandomForestClassifier | 0.973 | 0.991 | 0.982 |
| | LGBoost | 0.984 | 0.887 | 0.933 |
| | XGBClassifier | 0.975 | 0.987 | 0.981 |
| | CatBoostClassifier | 0.980 | 0.991 | 0.985 |

types of classifiers. In fact, the top three classifiers having the best F-measures belong to the ensemble methods namely CatBoost, Random Forest and XGBoost, while Naive Bayes performed worst with lowest two F-measures. A reason for this behavior is that ensemble methods are robust to overfitting as compared to Naive Bayes classifiers which tend to overfit the model. Thus, any method which overfits the model will suffer. Two classifiers are experimented in 'LDA and QDA' type. The F-measure of LDA is quite low, that is, 0.471 while the F-measure of QDA is observed to be 0.782 which shows a percent increase in the performance of 66% while the percent increase in the performance from Naive Bayes to 'LDA and QDA' type is 248%. The type 'Generalized Linear Models' performed no better than the type 'LDA and QDA'. The F-measure obtained from its classifier, that is, Logistic Regression, is 0.787. The percent increase in the performance from 'LDA and QDA' to the type of 'Generalized Linear Models' is only 0.64%. One classifier from each of SGD, SVM, DT, NN and Nearest Neighbors is tested. Their F-measures are observed as $0.945, 0.972, 0.977, 0.979$ and $0.980$, respectively. Notably, all these readings are above 0.90. LGBoost has the worst F-measure among ensemble methods which is 0.933 while the F-measure of AdaBoost is 0.971. The percent increase in the performance from the worst to the best classifier in the ensemble methods is only 5.5%,

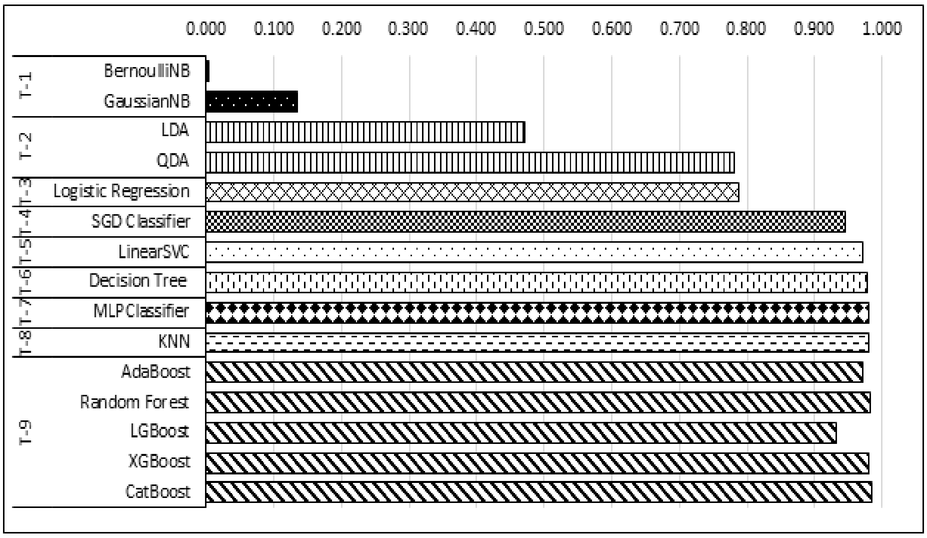

FIGURE 5.3: Comparison of Different Types of Classifiers using F-measure

which shows that the performance of all classifiers in ensemble methods is close to each other. The comparison of all types of classifiers using F-measure is shown in Figure 5.3 where T-1 to T-9 correspond to the types of Naive Bayes, LDA and QDA, Generalized Linear Models, SGD, SVM, DT, NN, Nearest Neighbors and Ensemble Methods, respectively.

Considering recall as the efficiency measure of the classifiers, NN outperformed other types with MLP Classifier having a recall of 0.994. The worst two recalls are observed for the Naive Bayes. For the type 'LDA and QDA', LDA has a recall as low as 0.337, while the recall of QDA is 0.829, which shows a performance increase of 146%. The type 'Generalized Linear Models' performed no better than 'LDA and QDA'. The classifier used for this type is Logistic Regression. Its recall is 0.663. An interesting point is that the counterpart of logistic regression, that is, MLP Classifier which belongs to the type of NN, has the highest recall. Thus, the percentage increase in performance from the Logistic Regression to MLP Classifier is 50%. The only difference between the two classifiers is the number of hidden layers between the input and the output

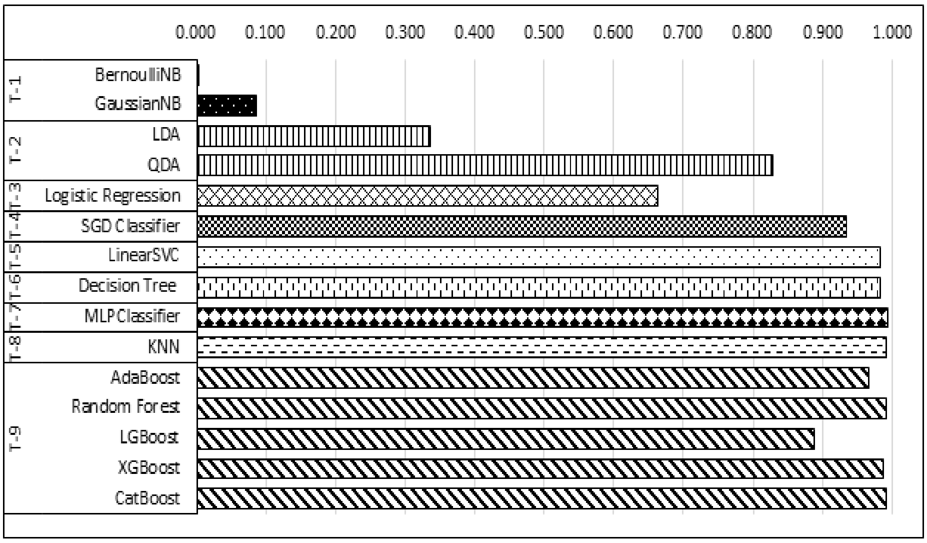

FIGURE 5.4: Comparison of Different Types of Classifiers using Recall

layer. This observation has led us to a new future direction of testing deep learning in our real dataset. The recalls of each of the classifiers from SGD, SVM, DT and Nearest Neighbors are observed as 0.936, 0.984, 0.984 and 0.993, respectively. The recalls of ensemble methods are 0.966, 0.991, 0.887, 0.987 and 0.991. These recalls are for the classifiers AdaBoost, Random Forest, LGBoost, XGBoost and CatBoost, respectively. Other than LGBoost, the recalls of all ensemble methods are above 0.960, which shows that the performance of ensemble methods is very good for our data. The percent increase from the worst to the best classifier in the ensemble methods is 11.7%. The comparison of different types of classifiers using recall is shown in Figure 5.4.

Considering precision as the efficiency measure of the classifiers, all the classifiers used in ensemble methods outperformed rest of the types. LGBoost has the best precision of 0.984, which interestingly also has the lowest recall and F-measure among the ensemble methods. This indicates that LGBoost has the lowest FP. The other classifiers of AdaBoost, Random Forest, XGBoost and CatBoost from ensemble methods have precision reading as 0.976, 0.973, 0.975 and

0.980, respectively. There is an increase of only 1.1% of the performance from the worst to the best classifier among the ensemble methods. This shows that not only all classifiers in this type performed better but also their performance is very close to each other. The worst two precisions are for the type 'Naive Bayes'. The two classifiers from the type 'LDA and QDA' has the precision of 0.782 and 0.740. This shows an increase of 48% in the performance from Naive Bayes type to 'LDA and QDA' type. Each of the classifiers from the types of Generalized Linear Models, SGD, SVM, DT, NN and Nearest Neighbors has precision readings as 0.968, 0.955, 0.960, 0.971, 0.964 and 0.967, respectively.

#### 5.3.1.2 Best Performing Classifiers for NTL Detection

We have used precision, recall and F-measure as the performance evaluation metrics. The best F-measure is 0.985 for CatBoost classifier, which narrowly outperforms Random Forest and KNN. These three classifiers have corresponding high precision and recall values indicating small FP and small FN values, respectively.

The F-measure of LGBoost classifier is 0.933, which is comparatively less than the F-measure of CatBoost classifier, i.e 0.985, while the corresponding figure for XGBoost is 0.981. There is an increase of 5.6% in the F-measure from LGBoost to CatBoost. Overall, precision and recall obtained for CabBoost, XGBoost and LGBoost classifiers are above 0.97 except that the recall of LGBoost is 0.887.

The F-measures is significantly increased from 0.471 to 0.782 when choosing QDA instead of LDA, which indicates that QDA outperforms LDA. This is because when multiple classes have a different co-variance relationship then LDA suffers while QDA remains a better option. This gives an insight to the characteristics of features of this real dataset, that is, for NTL, there is a room to explore more about the co-variance relationship for individual classes.

## 5.3.2 Performance Analysis of Deep Learning on Reduced Feature-Set

In addition to finding the best classifiers and the types of the classifiers for NTL detection, a separate experiment is performed on the

TABLE 5.3: TP, TN, FP, FN, Precision, Recall and F-Measure of Neural Networks with Different Layers

| **Layers** | **Units** | **TP** | **TN** | **FP** | **FN** | **Pre.** | **Recall** | **F-M** |
|---|---|---|---|---|---|---|---|---|
| 2 | 100 | 648 | 15352 | 14 | 35 | 0.979 | 0.949 | 0.964 |
| 3 | 100 | 683 | 15319 | 47 | 0 | 0.936 | 1.000 | 0.967 |
| 4 | 100 | 647 | 15349 | 17 | 36 | 0.974 | 0.947 | 0.961 |
| 5 | 100 | 676 | 15347 | 19 | 7 | 0.973 | 0.990 | 0.981 |
| 10 | 100 | 682 | 15328 | 38 | 1 | 0.947 | 0.999 | 0.972 |

same dataset with different layers of neural network (MLP) while each layer consists of 100 units. Five MLPs are tested with layers of $2, 3, 4, 5$, and 10, respectively. Other than different layers, all other parameters of MLP is kept same as presented in Appendix B. The TP, TN, FP and FN values of each MLP is presented in Table 5.3. As depicted in Figure 5.5, the maximum precision of 0.979 is observed for the MLP with 2 layers while the maximum recall of 1 is observed for the MLP with 3 layers. Considering recall, the MLP with 3 layers has outperformed all the other classifiers. The maximum F-measure of 0.981 is observed for the MLP with 5 layers which is comparable to the F-measures of the other classifiers. These results have encouraged us for experimenting an exploratory analysis of deep learning with different parameters in NTL detection using a real dataset.

## 5.4 Conclusion and Future Work

This contribution has used a real-world dataset of a power supply company in Pakistan for NTL detection. The dataset contains approximately $80,000$ monthly consumption records along with 71 features. We have tested 15 machine learning classifiers, which span across 9 types for a potential detection of NTL. The classifiers also include recently developed ensemble methods namely CatBoost, LGBoost and XGBoost. As the dataset belongs to the imbalance domain, we have used precision, recall and F-measure as the performance evaluation metric.

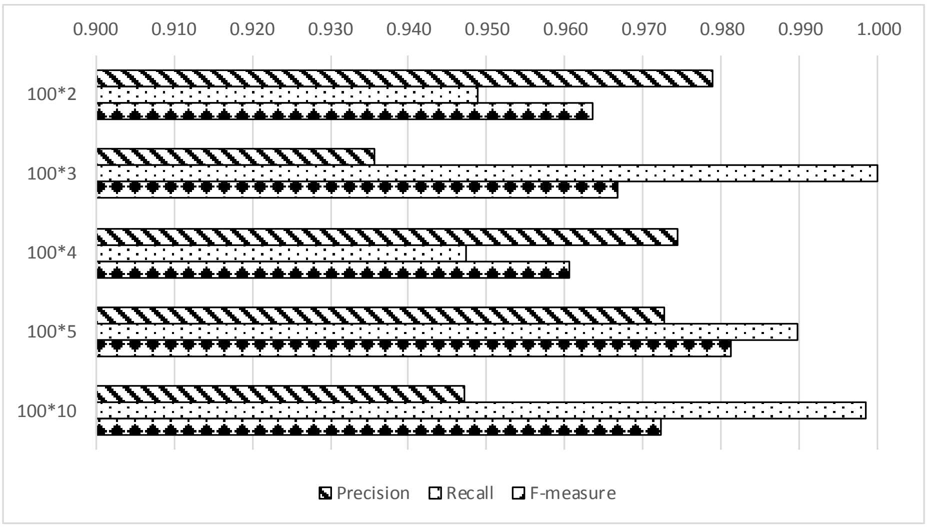

FIGURE 5.5: Precision, Recall and F-Measure of Neural Networks with Different Layers

One of our findings of this work is that, with respect to F-measure, ensemble methods outperformed other types and with respect to Recall, ANN outperformed other types of the classifiers. Considering individual classifier analysis, CatBoost outperformed all other classifiers when taking F-measure into account, while MLP Classifier performed best when considering Recall as the performance evaluation metric. One of the observations is that recall increases by 50% when MLP Classifier is used instead of Logistic Regression. This shows that testing deep learning with many hidden layers can be a potential future contribution in NTL detection. Another important observation is that not all the features in a real dataset are useful in detecting NTL. Using feature importance along with Gini Index, we have derived a mechanism to identify the top-14 features, out of 71 features, which are contributing 77% in NTL detection. This has not only significantly reduced the execution time but also has identified useful features for NTL detection in a real dataset. This analysis can be used as a baseline for future work to obtain the best combination of features in a real dataset considering all classifiers.

Another contribution of this work is that it has analyzed the results of all classifiers with respect to their types. As a result, it has outlined significant observations about the types of the classifiers with respect to NTL detection. This contribution has opened a new area for the research community working in NTL detection.

There is still a need for creating a benchmark dataset which can widely be used in NTL detection. Another future direction is using penalized machine learning models in which weighted classifiers (Awais et al., 2019) are used. The best classifiers identified in this study can also be implemented on different feature selection approaches.

# Chapter 6

# Impact of Feature Selection on Non-Technical Loss Detection

## 6.1 Introduction

An important consideration about the problem of NTL detection is that it belongs to imbalance domain. It is a specific type of problem where there is an imbalance in the distribution of classes. The normal consumers (negative class) are too many whereas the representation of the thieves (positive class) is too small. Interestingly, we are more focused in finding the instances of the least representative class (Awais, Palmerini, and Chiari, 2016), i.e. the positive class. This imbalanced distribution of classes has made the problem of NTL detection somewhat a unique problem. The research community has proposed solutions to deal with imbalance datasets like oversampling the minority class (Chawla et al., 2002) or random under-sampling of the majority class (Liu, Wu, and Zhou, 2008).

One of the most important ingredients in solving a machine learning problem is the data used to solve the problem. Not only relevant records are important for correctly predicting the class labels but identifying relevant features are equally important. For NTL detection, efforts have been made in identifying and representing relevant records by addressing the issue of class imbalance but no such focus has been made in identifying relevant features. In this section, we focus on identifying relevant features for NTL detection with respect to the target variable. Then, using those features we derive a mechanism to filter out features which are best suitable for different classifiers using our proposed Incremental Feature Selection

(IFS) algorithm. Our proposed methodology is based on four main steps: (1) data collection and pre-processing, (2) extraction of feature importance, (3) use of the IFS algorithm for classification given feature importance, and (4) use of performance evaluation metrics. Initially, the data contained 112 features out of which 71 useful features are shortlisted after pre-processing. Out of the 71 features, a further most relevant features for NTL detection are identified by the IFS algorithm.

In summary, we propose a novel solution for the selection of the most relevant features in a real dataset for different classifiers used for NTL detection in a bid to minimize the overall execution time of classification. Our contribution to the literature is the following:

1. We use feature importance to compare and order each feature with respect to its relevancy to the target variable. No such work has been done before for NTL detection.

2. We propose the Incremental Feature Selection (IFS) algorithm, which systematically adds new features and calculates performance evaluation metrics of different classifiers for NTL detection.

3. From a list of 71 features, we have identified 9 features having almost the same accuracy measures as was achieved with 71 features.

4. Unlike other techniques, we have used three evaluation metrics which are best suitable to evaluate the classifiers used for class imbalance domain.

## 6.2 Data Collection and Pre-Processing

For the experimentation of feature selection, we have used the same dataset as referred in Chapter 4 and 5. The summarized characteristics of this dataset is that it is used for a classification problem having class labels of 'Yes' or 'No' for theft and non-theft cases, respectively. The Class 'Yes' contains only 4% of the representation while the class 'No' contains 96% of the representation. This characteristic has made the real dataset a perfect case of class imbalance problem where representation of one class out-weights the other class. The data is then divided into training and test sets in the ratio of 80% : 20%, respectively. However, it was made sure that the original distribution ratio of theft and non-theft is retained in the training and the test set. A complete description of the dataset is presented in Chapter 3.

## 6.3 The Need for Feature Selection

A total of 71 features are initially shortlisted from the dataset of the electricity supplier. Some features are more important in predicting the theft cases while others remain irrelevant. Making the models learn on irrelevant features causes a decrease in the accuracy. Moreover, there is also an occurrence of duplicate features which causes overall performance deterioration. To overcome this problem and to find which features are more relevant in theft prediction, a feature selection technique is required. For this, we have used feature importance. It is a numerical score associated with each feature in the dataset. The higher the score, the more relevant is the feature to the target variable. A theoretical explanation of feature importance is described in (Breiman, 2001).

## 6.4 Machine Learning Classifiers

In order to predict the NTL in our real dataset, we have tested three machine learning classifiers namely CatBoost (Prokhorenkova et al., 2018), *k*-Nearest Neighbors (KNN) (Goldberger et al., 2005) and Decision Tree (DT) Classifier. The classifiers are selected due to their success in class imbalance problems. The details of these classifiers are presented in Section 2.2.2.

## 6.5 Evaluation Metric

The classifiers used in class imbalance problems have an interesting property that they can not be evaluated with the metrics that are used in normal classification problem. For example, accuracy is not the right measure for the evaluation of classifiers in imbalance domain as it is dominated by the number of non-theft cases correctly classified by the classifier (True Negative), which in our case is 96%. For this reason, we have chosen F-Measure as the evaluation metric, which uses precision and recall. The details of precision, recall and F-Measure are presented in Section 2.2.3.

## 6.6 The Incremental Feature Selection Algorithm

In this section, we present our Incremental Feature Selection (IFS) algorithm for selecting the best features across multiple classifiers. The IFS is listed in Algorithm 1. The set of selected features are first sorted in reverse order with respect to the Gini index value (Raileanu and Stoffel, 2004) (feature importance). A theoretical description of Gini Index can be found in Section 5.2.3. This is shown in step 1 of the IFS algorithm. As a result, the feature with the highest information gain comes in the first place, the feature with the second highest information gain comes in the second place, and so on. The list of classifiers is stored in $List\_of\_classifiers$. A new list, $f\_selected$, which is initially empty, is incrementally appended by adding the next feature from the $List\_of\_features$. For every new feature set, the data is split into training and test set. Once the training and the test sets are ready for the features in $f\_selected$, we do the following for every classifier in $List\_of\_classifiers$. First, train the classifier with the training set. Second, test the classifier with the test set. Third, compute the confusion matrix containing $TP$, True Negative ($TN$), False Positive ($FP$) and $FN$ values. Finally, compute precision, recall and F-Measure. After complete execution of the algorithm, the F-Measure for the selected classifiers is obtained with initially only first feature, then with first two features, and so on. It continues until all features from $List\_of\_features$ are selected.

### 6.6.1 Early Stopping

In our simulation, we have set an early stopping criterion for the IFS algorithm which is when the difference of performance metrics using selected features, as compared to all features, is reduced to −1% or less, the algorithm terminates without adding new features to the list.

**Algorithm 1** Incremental Feature Selection (IFS) algorithm

```
Input: $List\_of\_features, Gini\_index, List\_of\_classifiers$
Output: $precision, recall, fmeasure$ {2-D arrays having accuracy values}
 1: $fs \leftarrow reverse\_sort(List\_of\_features, Gini\_index)$
 2: $f\_selected \leftarrow [\,]$
 3: $precision \leftarrow [\,][\,]$
 4: $recall \leftarrow [\,][\,]$
 5: $fmeasure \leftarrow [\,][\,]$
 6: for $f$ in $fs$ do
 7:     f_selected.append ($f$)
 8:     $x\_train, y\_train, x\_test, y\_test \leftarrow split(f\_selected)$
 9:     for $c$ in $List\_of\_classifiers$ do
10:         $model \leftarrow train\_classifier(c, x\_train, y\_train)$
11:         $y\_predict \leftarrow test\_classifier(model, x\_test)$
12:         $TP, FP, FN, TN \leftarrow confusion\_matrix(y\_test, y\_predict)$
13:         $precision[c, f\_selected] \leftarrow TP/(TP + FP)$
14:         $recall[c, f\_selected] \leftarrow TP/(TP + FN)$
15:         $fmeasure[c, f\_selected] \leftarrow \frac{2 \times precision[c,f\_selected] \times recall[c,f\_selected]}{precision[c,f\_selected] + recall[c,f\_selected]}$
16:     end for
17: end for
18: return $precision, recall, fmeasure$
```

### 6.6.2 Experimental Setup

The simulation of the IFS algorithm was carried out with Python's open source library scikit-learn (Pedregosa et al., 2011b) using a 64-bit Windows server with Intel Xeon 2.2 GHz processor and 32 GB RAM. The learning rate for CatBoost was set to 0.047 and the logloss

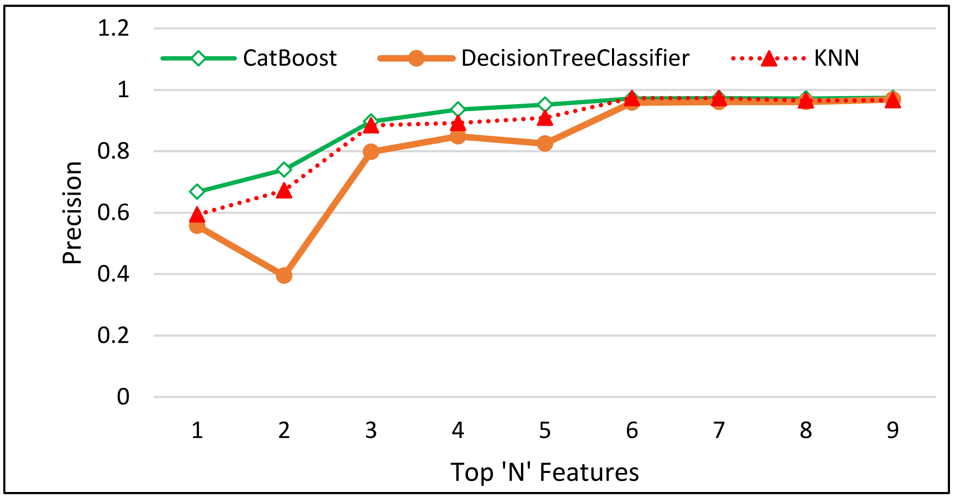

FIGURE 6.1: Precision of CatBoost, KNN and Decision Tree Classifier with 9 Features

function was used as a loss function. Gini was used as a measure of information gain for the Decision Tree Classifier. For KNN, $K$ was set to 5 and the distance metric used was Minkowski.

## 6.7 Results and Discussions

The performance of the proposed framework has been analyzed by computing precision, recall and F-Measure of recently developed CatBoost (Prokhorenkova et al., 2018), KNN (Goldberger et al., 2005) and Decision Tree Classifier (Mingers, 1989). Figure 6.1 shows the precision, Figure 6.2 shows the recall and Figure 6.3 shows the F-Measure of the three classifiers. The figures represent the top-9 features as identified by the IFS algorithm. These features are incrementally added with respect to their feature importance.

A significant increase in overall performance is observed for all classifiers with the inclusion of the $3^{rd}$ and $6^{th}$ features. The total number of units consumed in current year is stored in the $3^{rd}$ feature, while the $6^{th}$ feature records average amount paid during the last 12 months. Performance of the Decision Tree Classifier is boosted with the addition of the $3^{rd}$ feature, while the $6^{th}$ feature has a significant

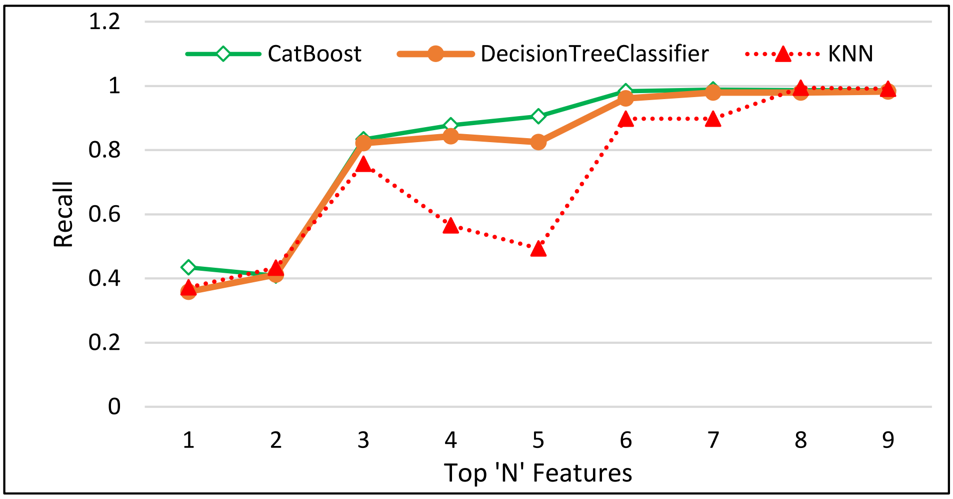

FIGURE 6.2: Recall of CatBoost, KNN and Decision Tree Classifier with 9 Features

impact on recall and F-Measure of KNN. With the inclusion of the 9th feature, the precision, recall and F-Measure of the three classifiers achieved near optimum values which are comparative to the corresponding figures when all features are included.

TABLE 6.1: Precision, Recall and F-Measure of CatBoost, Decision Tree Classifier and KNN for 9 and 71 features

| | Features | CatBoost | DT | KNN |
|---|---|---|---|---|
| Precision | 71 | 98.11% | 97.23% | 94.18% |
| | 9 | 97.40% | 96.83% | 96.58% |
| Recall | 71 | 99.27% | 97.80% | 45.10% |
| | 9 | 98.68% | 98.24% | 99.12% |
| F-Measure | 71 | 98.69% | 97.51% | 61.00% |
| | 9 | 98.04% | 97.53% | 97.83% |

An interesting observation about the performance of the three classifiers is that it is comparative or slightly better when the features are selected using the IFS algorithm as compared to when all the features are selected. This can be verified in Table 6.1, where selecting the top 9 features using the IFS algorithm, the precision,

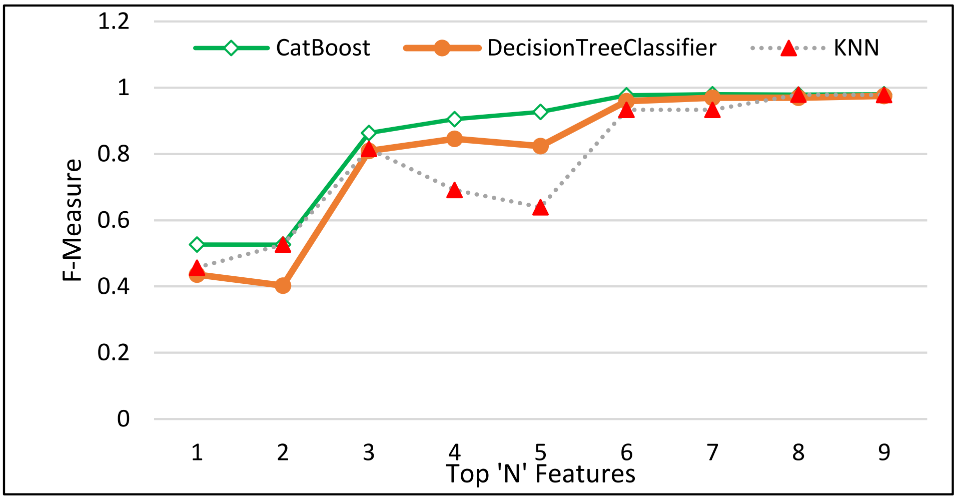

FIGURE 6.3: F-Measure of CatBoost, KNN and Decision Tree Classifier with 9 Features

TABLE 6.2: Execution Time of CatBoost, Decision Tree Classifier and KNN for 9 and 71 features

| | Features | CatBoost | DT | KNN |
|---|---|---|---|---|
| Training Time | 71 | 149s | 7s | 5s |
| | 9 | 94s | 2s | 0.5s |
| Percentage Save-Up Time | | 37% | 71% | 90% |

recall and F-Measure achieved their near optimum. In some cases, the performance with selected features using the IFS algorithm is slightly better than the performance achieved using all features. Table 6.1 depicts that performance of CatBoost is slightly deteriorated with the use of the IFS algorithm. Precision and F-Measure of CatBoost using the IFS algorithm is decreased by 1% while recall remained the same. On the other hand, there is a major decrease in the training time of CatBoost with the use of the IFS algorithm as shown in Table 6.2. The training time of CatBoost with all features included is 149 sec and with the use of the IFS algorithm, it is reduced to 94 sec, which is a 37% reduction. Analyzing the three metrics, one can observe that the performance of KNN is improved significantly using the IFS algorithm. For KNN, precision is increased by 3%, while recall and F-Measure are increased by 120% and 60%, respectively. The training time of KNN using the IFS algorithm is also reduced by 90%. For decision tree, the performance remains almost the same with the use of the IFS algorithm. Its precision is decreased by 0.5%, recall is increased by 0.5%, while F-Measure remained the same but the training time is reduced by 71%. This shows that the IFS algorithm has not only identified key features which are participating in NTL detection but it has also significantly reduced the overall training time of classifiers.

# Chapter 7

# Conclusions and Future Work

## 7.1 Conclusions

This dissertation mainly focused on the problem of NTL detection on a real dataset. The dataset is taken from a power supply company in Pakistan. The dataset contains approximately $80,000$ monthly consumption records during a span of January 2015 to March 2016.

The objective of the dissertation was to first identify the open challenges and the knowledge gap in NTL detection through extensive literature review and then address those challenges by applying recently developed tools and techniques in a real dataset.

The contributions of the dissertation are listed below:

1. We presented a comprehensive literature review of NTL. This review is presented in Table 2.1 of Chapter 2.

2. Based on the literature review, we identified the limitations of recent work in NTL detection. The limitations are outlined below:

   (a) There is a need to identify the suitable performance evaluation metrics for NTL detection considering the specific requirements of NTL detection.

   (b) The identification of the types of the classifiers which are best suited for NTL detection in a real dataset.

   (c) The identification of the individual classifiers which are best suited for NTL detection in a real dataset.

   (d) It is highly likely that not all the features are good in participating in the detection of NTL. So, there is a need to identify the most relevant features in a real dataset that can be used in the identification of NTL.

These limitations and gaps identified in the literature review were addressed by the identification of performance evaluation metrics that are best suitable for NTL detection, the comparison of the classifiers and the types of the classifiers for NTL detection and the impact of feature selection in NTL detection.

3. The problem of NTL detection should be dealt with considerations that the number of fraudulent attempts are negligible as compared to the number of normal consumption. This means that the problem belongs to the class imbalance domain. This biasness in the class distribution emphasises on the point that care must be taken in not only applying the machine learning classifiers but also in the performance evaluation metrics that are used in identifying potential NTL. The dissertation contributes in analyzing 14 performance evaluation metrics and identifying the suitable ones for NTL detection. The datasets that are used in the problem of NTL detection have a high imbalanced ratio of the positive class and the negative class. Adding the importance of finding True Positives (TP), False Negatives (FN) and False Positives (FP) to the problem of NTL detection, it is necessary to first identify the requirements of NTL detection and then select the best metrics for performance evaluation while taking care of the specific requirements of NTL detection. One of our findings is that precision and recall should not be given equal priority for NTL detection. Rather, recall should be given higher priority. We have tested KNN, random forest and linear SVM classifiers for the identification of NTL attempts. The percent increase of recall from using KNN to random forest was observed as 1.24%. So, we concluded that random forest is the better choice as it has the highest recall. This analysis can be used as a baseline for the accurate selection of the classifiers in NTL detection. A complete description of the contribution is presented in Chapter 4.

4. The dissertation contributes in testing 15 machine learning classifiers which belong to 9 different types of the classifiers which also includes the recently developed CatBoost, LGBoost and XGBoost classifiers. In this contribution, not only individual classifiers are compared for NTL detection but the types of the classifiers are also compared. As a result, it has outlined significant observations about the types of the classifiers with respect to NTL detection. Based on the findings of the best evaluation metric in our previous contribution, the MLP classifier was found as the best individual classifier with respect to recall and ANN was found as the best type of the classifiers for NTL detection. A complete description of the contribution is presented in Chapter 5.

5. The dissertation proposes a novel framework to identify relevant features for NTL detection. This framework is composed of our proposed Incremental Feature Selection (IFS) algorithm, which first identifies the most relevant features for NTL detection in a real dataset using feature importance. Then, it incrementally adds features with respect to their scores of importance to test the listed classifiers. Finally, it calculates performance evaluation metrics for the selected classifiers. In this contribution, CatBoost, Decision Tree Classifier and KNN are tested for NTL detection due to their success in imbalance domain. The results have shown that with the use of the IFS algorithm, recall and F-Measure of KNN is increased by 120% and 60%, respectively, while the training time of KNN is reduced by 90%. The performance achieved using a shortlisted set of features by the IFS algorithm is comparable to the performance achieved using all the features of the dataset. Moreover, this framework has also considerably minimized the execution time of all the classifiers. A complete description of the contribution is present in Chapter 6.

## 7.2 Future Work

There is a need to further extend the use of performance evaluation metrics that can estimate and compare error rates on the basis of which a combination of classifiers can be selected for a specified dataset for NTL detection. Currently, there is a small range of graphical metrics used for performance analysis. This includes receiver operating characteristic (ROC) and area under ROC curve (AUC). There is also a need of further exploration in the usage of graphical performance metrics.

The problem of NTL detection is a global problem. In order to provide the research community a chance to further extend their contributions in this field, there is a need of a benchmark dataset which can widely be used in NTL detection. Due to the highly sensitive nature of the data, currently no such benchmark dataset is available. There is a possible future direction of the creation of a real as well as a synthetic dataset for the problem of NTL detection. Another related future direction can be a comparison of the results achieved in this study with the experimental results performed on a different real or synthesized dataset.

Another future direction is using penalized machine learning models in which weighted classifiers are used. The best classifiers identified in this study can also be implemented on different feature selection approaches.

Moreover, with the amount of success deep learning has been receiving in the last few years, there is also a need to test the performance of deep learning in NTL detection using selected feature set and all feature set. Moreover, deep learning can also be tested in feature selection for a real dataset. Over sampling and under sampling techniques have shown good results in many problems. NTL detection can also be thoroughly tested with over sampling and under sampling techniques with a real dataset.

Power industries face a closely related problem of defaulters. The defaulters do not pay the bill for many months. The dataset used in NTL detection can also be used in the prediction of defaulters. Rather, an interesting fact finding research activity can be proceeded which finds the co-relation between the defaulters and the NTL fraudsters.

The datasets used in NTL detection can also be merged with some external data representing the features of environment like temperature, humidity and climate readings. This way a seasonal comparison of NTL occurrences can also be made. This work can further be extended for the seasonal comparison of NTL with the seasonal comparison of defaulters. This way, the impact of seasonal changes on the occurrences of NTL and defaulters prediction can be measured.

# Bibliography

Alam, MS et al. (2004). "Power sector reform in Bangladesh: Electricity distribution system". In: *Energy* 29.11, pp. 1773–1783.

Altman, Naomi S (1992). "An introduction to kernel and nearest-neighbor nonparametric regression". In: *The American Statistician* 46.3, pp. 175–185.

Amalina, F. et al. (2019). "Blending Big Data Analytics: Review on Challenges and a Recent Study". In: *IEEE Access*, pp. 1–1.

Ariyaluran Habeeb, Riyaz Ahamed et al. (2019). "Clustering-based real-time anomaly detection—A breakthrough in big data technologies". In: *Transactions on Emerging Telecommunications Technologies* 0.0, e3647.

Avila, Nelson Fabian, Gerardo Figueroa, and Chia-Chi Chu (2018). "NTL Detection in Electric Distribution Systems Using the Maximal Overlap Discrete Wavelet-Packet Transform and Random Undersampling Boosting". In: *IEEE Transactions on Power Systems* 33.6, pp. 7171–7180.

Awais, Muhammad, Nasreen Badruddin, and Micheal Drieberg (2017). "A hybrid approach to detect driver drowsiness utilizing physiological signals to improve system performance and wearability". In: *Sensors* 17.9, p. 1991.

Awais, Muhammad, Luca Palmerini, and Lorenzo Chiari (2016). "Physical activity classification using body-worn inertial sensors in a multi-sensor setup". In: *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. IEEE, pp. 1–4.

Awais, Muhammad et al. (2019). "An Internet of Things based bed-egress alerting paradigm using wearable sensors in elderly care environment". In: *Sensors* 19.11, p. 2498.

Balakrishnama, Suresh and Aravind Ganapathiraju (1998). "Linear discriminant analysis-a brief tutorial". In: *Institute for Signal and information Processing* 18, pp. 1–8.

Bhat, Rajendra Rana et al. (2016). "Identifying nontechnical power loss via spatial and temporal deep learning". In: *Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on*. IEEE, pp. 272–279.

Branco, Paula, Luís Torgo, and Rita P. Ribeiro. "A Survey of Predictive Modeling on Imbalanced Domains". In: *ACM Comput. Surv.* 49.2 (), 31:1–31:50. ISSN: 0360-0300.

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Buevich, Maxim et al. (2016). "Microgrid Losses: When the whole is Greater than the Sum of its parts". In: *Proceedings of the 7th International Conference on Cyber-Physical Systems*. IEEE Press, pp. 46–50.

Cao, Li-Juan and Francis Eng Hock Tay (2003). "Support Vector Machine with Adaptive Parameters in Financial Time Series Forecasting". In: *IEEE Transactions on neural networks* 14.6, pp. 1506–1518.

Chang, Chih-Chung and Chih-Jen Lin (May 2011). "LIBSVM: A Library for Support Vector Machines". In: *ACM Trans. Intell. Syst. Technol.* 2.3, 27:1–27:27. ISSN: 2157-6904.

Chatterjee, Soham et al. (2017). "Detection of non-technical losses using advanced metering infrastructure and deep recurrent neural networks". In: *Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe), 2017 IEEE International Conference on*. IEEE, pp. 1–6.

Chauhan, A Abhishek (2015). "Non-technical losses in power system and monitoring of electricity theft over low-tension poles". In: *Advances in Computing and Communication Engineering (ICACCE), 2015 Second International Conference on*. IEEE, pp. 280–284.

Chawla, Nitesh V et al. (2002). "SMOTE: synthetic minority oversampling technique". In: *Journal of artificial intelligence research* 16, pp. 321–357.

Chen, Tianqi and Carlos Guestrin (2016). "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* . ACM, pp. 785–794.

Cody, Christa, Vitaly Ford, and Ambareen Siraj (2015). "Decision Tree Learning for Fraud Detection in Consumer Energy Consumption." In: *ICMLA*, pp. 1175–1179.

Coma-Puig, Bernat et al. (2016). "Fraud detection in energy consumption: a supervised approach". In: *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on Data Science*. IEEE, pp. 120–129.

Di Martino, Matías et al. (2012). "Improving Electric Fraud Detection using Class Imbalance Strategies". In: *ICPRAM (2)*, pp. 135–141.

Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin (2018). "CatBoost: gradient boosting with categorical features support". In: *arXiv pre- print arXiv:1810.11363*.

Figueroa, Gerardo et al. (2017). "Improved practices in machine learning algorithms for NTL detection with imbalanced data". In: *Power & Energy Society General Meeting, 2017 IEEE*. IEEE, pp. 1–5.

Ford, Vitaly, Ambareen Siraj, and William Eberle (2014). "Smart grid energy fraud detection using artificial neural networks". In: *Computational Intelligence Applications in Smart Grid (CIASG), 2014 IEEE Symposium on*. IEEE, pp. 1–6.

Freund, Yoav and Robert E Schapire (1997). "A decision-theoretic generalization of on-line learning and an application to boosting". In: *Journal of computer and system sciences* 55.1, pp. 119–139.

García, Vicente, Ramón A Mollineda, and J Salvador Sánchez (2008). "A New Performance Evaluation Method for two-class Imbalanced Problems". In: *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition(SSPR)*. Springer, pp. 917 –925.

Ghori, Khawaja Moyeezullah et al. (2019). "Performance analysis of different types of machine learning classifiers for non-technical loss detection". In: *IEEE Access* 8, pp. 16033–16048.

Ghori, Khawaja MoyeezUllah et al. (2020a). "Impact of Feature Selection on Non-technical Loss Detection". In: *2020 6th Conference on Data Science and Machine Learning Applications (CDMA)*. IEEE, pp. 19–24.

Ghori, Khawaja MoyeezUllah et al. (2020b). "Performance analysis of machine learning classifiers for non-technical loss detection". In: *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16.

Glauner, Patrick et al. (2016a). "Large-Scale Detection of Non - Technical Losses in Imbalanced Datasets". In: *Innovative Smart Grid Technologies Conference (ISGT), 2016 IEEE Power & Energy Society*. IEEE, pp. 1–5.

Glauner, Patrick O. et al. (2016b). "The Challenge of Non-Technical Loss Detection using Artificial Intelligence: A Survey". In: *CoRR* abs/1606.00626.

Goldberger, Jacob et al. (2005). "Neighbourhood components analysis". In: *Advances in neural information processing systems*, pp. 513–520.

Guerrero, Juan Ignacio et al. (2018). "Non-Technical Losses Reduction by Improving the Inspections Accuracy in a Power Utility". In: *IEEE Transactions on Power Systems* 33.2, pp. 1209–1218.

Han, Jiawei, Micheline Kamber, and Jian Pei (2011). "Data mining concepts and techniques third edition". In: *Morgan Kaufmann*.

Han, Wenlin and Yang Xiao (2014). "NFD: a practical scheme to detect non-technical loss fraud in smart grid". In: *Communications (ICC), 2014 IEEE International Conference on*. IEEE, pp. 605–609.

— (2019). "Edge computing enabled non-technical loss fraud detection for big data security analytic in Smart Grid". In: *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–12.

Hartmann, Thomas et al. (2015). "Suspicious electric consumption detection based on multi-profiling using live machine learning". In: *Smart Grid Communications (SmartGridComm), 2015 IEEE International Conference on*. IEEE, pp. 891–896.

Hayat, M. K. et al. (2019). "Towards Deep Learning Prospects: Insights for Social Media Analytics". In: *IEEE Access* 7, pp. 36958–36979.

Hearst, Marti A. et al. (1998). "Support Vector Machines". In: *IEEE Intelligent Systems and their applications* 13.4, pp. 18–28.

Ho, Tin Kam (1995). "Random decision forests". In: *Document analysis and recognition, 1995., proceedings of the third international conference on*. Vol. 1. IEEE, pp. 278–282.

Hussain, Zahoor et al. (2016). "Methods and techniques of electricity thieving in Pakistan". In: *Journal of Power and Energy Engineering* 4, pp. 1–10.

Jain, Ankit Kumar and Brij B Gupta (2019). "A machine learning based approach for phishing detection using hyperlinks information". In: *Journal of Ambient Intelligence and Humanized Computing* 10.5, pp. 2015–2028.

John, George H and Pat Langley (1995). "Estimating continuous distributions in Bayesian classifiers". In: *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., pp. 338–345.

Júnior, Leandro Aparecido Passos et al. (2016). "Unsupervised nontechnical losses identification through optimum-path forest". In: *Electric Power Systems Research* 140, pp. 413–423.

Ke, Guolin et al. (2017). "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in Neural Information Processing Systems*, pp. 3146–3154.

Lakshmi, KS and G Vadivu (2019). "A novel approach for disease comorbidity prediction using weighted association rule mining". In: *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–8.

León, Carlos et al. (2011). "Variability and trend-based generalized rule induction model to NTL detection in power companies". In: *IEEE Transactions on Power Systems* 26.4, pp. 1798–1807.

Liaw, Andy, Matthew Wiener, et al. (2002a). "Classification and Regression by Random Forest". In: *R news* 2.3, pp. 18–22.

— (2002b). "Classification and regression by randomForest". In: *R news* 2.3, pp. 18–22.

Liu, Xu-Ying, Jianxin Wu, and Zhi-Hua Zhou (2008). "Exploratory undersampling for class-imbalance learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2, pp. 539–550.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press. ISBN: 0521865719, 9780521865715.

McCallum, Andrew, Kamal Nigam, et al. (1998). "A comparison of event models for naive bayes text classification". In: *AAAI-98 workshop on learning for text categorization*. Vol. 752. 1. Citeseer, pp. 41–48.

McDaniel, P. and S. McLaughlin (2009). "Security and Privacy Challenges in the Smart Grid". In: *IEEE Security Privacy* 7.3, pp. 75–77.

Meira, Jorge Augusto et al. (2017). "Distilling provider-independent data for general detection of non-technical losses". In: *Power and Energy Conference at Illinois (PECI), 2017 IEEE*. IEEE, pp. 1–5.

Messinis, George M and Nikos D Hatziargyriou (2018). "Review of non-technical loss detection methods". In: *Electric Power Systems Research* 158, pp. 250–266.

Mingers, John (Nov. 1989). "An Empirical Comparison of Pruning Methods for Decision Tree Induction". In: *Machine Learning* 4.2, pp. 227–243. ISSN: 1573-0565.

Mutupe, RM et al. (2017). "Electricity theft detection system with RF communication between distribution and customer usage". In: *PowerAfrica, 2017 IEEE PES*. IEEE, pp. 566–572.

Nagi, Jawad et al. (2010). "Nontechnical Loss Detection for Metered Customers in Power Utility using Support Vector Machines". In: *IEEE transactions on Power Delivery* 25.2, pp. 1162–1171.

Ng, Andrew Y (2004). "Feature selection, L 1 vs. L 2 regularization, and rotational invariance". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM, p. 78.

Papadimitriou, Christina et al. (2017). "Non-technical losses: detection methods and regulatory aspects overview". In: *CIRED-Open Access Proceedings Journal* 2017.1, pp. 2830–2832.

Pedregosa, Fabian et al. (2011a). "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.

— (2011b). "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.

Peng, Bo et al. (2016). "A two-stage pattern recognition method for electric customer classification in smart grid". In: *Smart Grid Communications (SmartGridComm), 2016 IEEE International Conference on*. IEEE, pp. 758–763.

Prokhorenkova, Liudmila et al. (2018). "CatBoost: unbiased boosting with categorical features". In: *Advances in Neural Information Processing Systems*, pp. 6639–6649.

Punmiya, Rajiv and Sangho Choe (2019). "Energy Theft Detection Using Gradient Boosting Theft Detector With Feature Engineering - Based Preprocessing". In: *IEEE Transactions on Smart Grid* 10.2, pp. 2326–2329.

Raileanu, Laura Elena and Kilian Stoffel (2004). "Theoretical comparison between the gini index and information gain criteria". In: *Annals of Mathematics and Artificial Intelligence* 41.1, pp. 77–93.

Ramos, Caio César Oba et al. (2011a). "New insights on nontechnical losses characterization through evolutionary-based feature selection". In: *IEEE Transactions on Power Delivery* 27.1, pp. 140–146.

Ramos, Caio CO et al. (2011b). "A novel algorithm for feature selection using harmony search and its application for non-technical losses detection". In: *Computers & Electrical Engineering ..* 37.6, pp. 886 –894.

Ramos, Caio CO et al. (2016). "On the study of commercial losses in Brazil: a binary black hole algorithm for theft characterization". In: *IEEE Transactions on Smart Grid* 9.2, pp. 676–683.

Raza, Mohsin et al. (2019). "Diagnosis and Monitoring of Alzheimer's Patients Using Classical and Deep Learning Techniques". In: *Expert Systems with Applications*.

Razzak, Muhammad Imran, Muhammad Imran, and Guandong Xu (Mar. 2019). "Big data analytics for preventive medicine". In: *Neural Computing and Applications*. ISSN: 1433-3058.

"Real-time big data processing for anomaly detection: A Survey" (2019). In: *International Journal of Information Management* 45, pp. 289 –307. ISSN: 0268-4012.

Rehman, Muhammad Habib ur et al. (2018). "Big data analytics in industrial IoT using a concentric computing model". In: *IEEE Communications Magazine* 56.2, pp. 37–43.

Rehman, Muhammad Habib ur et al. (2019). "The role of big data analytics in industrial Internet of Things". In: *Future Generation Computer Systems* 99, pp. 247–259.

Rish, Irina et al. (2001). "An empirical study of the naive Bayes classifier". In: *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3. 22, pp. 41–46.

Rumelhart, David E, Geoffrey E Hinton, Ronald J Williams, et al. (1988). “Learning representations by back-propagating errors”. In: *Cognitive modeling* 5.3, p. 1.

Saeed, Zafar et al. (June 2019). “What’s Happening Around the World ? A Survey and Framework on Event Detection Techniques on Twitter”. In: *Journal of Grid Computing* 17.2, pp. 279–312. ISSN: 1572-9184.

Sánchez-Zuleta, Carmen Cecilia, Juan Pablo Fernández-Gutiérrez, and Carlos César Piedrahita-Escobar (2017). “Identification of the characteristics incident to the detection of non-technical losses for two Colombian energy companies”. In: *Revista Facultad de Ingeniería Universidad de Antioquia* 84, pp. 60–71.

Sharma, Desh Deepak and SN Singh (2015). “Aberration detection in electricity consumption using clustering technique”. In: *International Journal of Energy Sector Management* 9.4, pp. 451–470.

Sharma, Desh Deepak et al. (2017). “Identification and characterization of irregular consumptions of load data”. In: *Journal of Modern Power Systems and Clean Energy* 5.3, pp. 465–477.

Singh, Shailendra and Abdulsalam Yassine (2018). “Big data mining of energy time series for behavioral analytics and energy consumption forecasting”. In: *Energies* 11.2, p. 452.

Spirić, Josif V, Slobodan S Stanković, and Miroslav B Dočić (2018). “Identification of suspicious electricity customers”. In: *International Journal of Electrical Power & Energy Systems* 95, pp. 635–643.

Srivastava, Santosh, Maya R Gupta, and Béla A Frigyik (2007). “Bayesian quadratic discriminant analysis”. In: *Journal of Machine Learning Research* 8.Jun, pp. 1277–1305.

Sun, Yanmin et al. (2007). “Cost-Sensitive Boosting for Classification of Imbalanced Data”. In: *Pattern Recognition* 40.12, pp. 3358–3378.

Terciyanli, Erman et al. (2017). “Score based non-technical loss detection algorithm for electricity distribution networks”. In: *Smart Grid and Cities Congress and Fair (ICSG), 2017 5th International Istanbul*. IEEE, pp. 180–184.

Tsang, S. et al. (2011). “Decision Trees for Uncertain Data”. In: *IEEE Transactions on Knowledge and Data Engineering* 23.1, pp. 64–78.

Vapnik, Vladimir (1998). *Statistical Learning Theory. 1998*. Wiley, New York.

Vapnik, Vladimir Naumovich (1999). "An Overview of Statistical Learning Theory". In: *IEEE Transactions on Neural Networks* 10.5, pp. 988–999.

Viegas, Joaquim L, Paulo R Esteves, and Susana M Vieira (2018). "Clustering-based novelty detection for identification of non - technical losses". In: *International Journal of Electrical Power & Energy Systems* 101, pp. 301–310.

Xia, Xiaofang et al. (2015). "A difference-comparison-based approach for malicious meter inspection in neighborhood area smart grids". In: *Communications (ICC), 2015 IEEE International Conference on*. ieee, pp. 802–807.

Yang, Xin-She et al. (2008). "Firefly algorithm". In: *Nature-inspired metaheuristic algorithms* 20, pp. 79–90.

Yeckle, Jaime and Bo Tang (2018). "Detection of Electricity Theft in Customer Consumption Using Outlier Detection Algorithms". In: *Data Intelligence and Security (ICDIS), 2018 1st International Conference on*. IEEE, pp. 135–140.

Zhang, Harry (2004a). "The optimality of naive Bayes". In: *AA* 1.2, p. 3.

Zhang, Tong (2004b). "Solving large scale linear prediction problems using stochastic gradient descent algorithms". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM, p. 116.

Zheng, Kedi et al. (2017). "Electricity theft detecting based on density-clustering method". In: *Innovative Smart Grid Technologies- Asia (ISGT-Asia), 2017 IEEE*. IEEE, pp. 1–6.

Zheng, Zibin et al. (2018). "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids". In: *IEEE Transactions on Industrial Informatics* 14.4, pp. 1606–1615.

Zhou, Gang et al. (2014). "A novel load profiling method for detecting abnormalities of electricity customer". In: *PES General Meeting | Conference & Exposition, 2014 IEEE*. IEEE, pp. 1–5.

Zou, Hui and Trevor Hastie (2005). "Regularization and variable selection via the elastic net". In: *Journal of the royal statistical society: series B (statistical methodology)* 67.2, pp. 301–320.

# Appendix A

# Feature Set

The list of 71 features along with their categories, feature description and feature importance values is described in Table A.1.

TABLE A.1: Feature Set

| FID | Feature | Description | Category | F. Importance |
|---|---|---|---|---|
| 1 | Units-12Months | Units consumed during last 12 months | Normal units | 0.141807732008 |
| 2 | Amount-12Months | Total amount billed during last 12 months | Normal amount | 0.116774856878 |
| 3 | BilledUnit-YTD | Units billed in current year | Normal units | 0.116751907486 |
| 4 | BilledAmount | Amount billed in current month | Normal amount | 0.092791430138 |
| 5 | 1 year LPS | Late payment surcharge in last one year | Additional amount | 0.057126632724 |
| 6 | Amount-12MonthsAvg | Average monthly amount in last 12 months | Normal amount | 0.040861765397 |
| 7 | BilledAmount-12MonthsGross | Total payment made in last 12 months | Normal amount | 0.038509950395 |
| 8 | Units-12MonthsAvg | Average monthly units in last 12 months | Normal units | 0.032045710111 |
| 9 | Amount-GrossBilledYTD | Total payment made in current year till date | Normal amount | 0.029088274439 |
| 10 | Amount-12MonthsAvgGrossBilled | Total average monthly payment made in last 12 months | Normal amount | 0.028248805369 |
| 11 | Amount-Regular | Payable amount for regular units | Normal amount | 0.023397470434 |
| 12 | 1 Month LPS | Late payment surchare in last 30 days | Additional amount | 0.022879096842 |
| 13 | Month-Billing | Month of billing | Bill info | 0.016062708402 |
| 14 | InstallementNo | Number of installements | Additional amount | 0.015831709285 |
| 15 | Units-Regular | Sum of all units | Normal units | 0.015017339757 |
| 16 | Billing-MonthYear | Month and year of billing | Bill info | 0.014896074379 |
| 17 | 6-12MonthLPS | Late payment surcharge in last 6-12 months | Additional amount | 0.013971961284 |
| 18 | 3-6MonthLPS | Late payment surcharge in last 3-6 months | Additional amount | 0.012784735142 |
| 19 | Billed-LPS | Late payment surcharge billed | Additional amount | 0.011929939987 |
| 20 | Amount-Installement | Amount of installements | Additional amount | 0.011181126362 |
| 21 | 2-3MonthLPS | Late payment surcharge in last 2-3 months | Additional amount | 0.010289314501 |
| 22 | Bill Day | Day of billing | Bill info | 0.009823534830 |
| 23 | 1-2MonthLPS | Late payment surchrge in last 1-2 months | Additional amount | 0.009627897668 |
| 24 | Balance-Closing | Current month bill dues | Normal amount | 0.009071362880 |
| 25 | Date-Last Payment | Date of last payment received | Bill info | 0.008636211093 |
| 26 | Balance-Opening | Previous month bill dues | Normal amount | 0.008609312214 |
| 27 | Date-Last Disconnection | Date of last disconnection | Extra info | 0.006633772523 |
| 28 | 1-2YearLPS | Late payment surcharge in last 1-2 years | Additional amount | 0.006118613036 |
| 29 | 2-3YearLPS | Late payment surcharge in last 2-3 years | Additional amount | 0.005575965464 |
| 30 | Class-Bill | Bill class as per type of customer | Bill info | 0.005432932386 |
| 31 | Amount-Last Payment | Amount of last payment received | Normal amount | 0.005270394928 |
| 32 | Meter Company | Meter manufacturer company | Extra info | 0.004691091511 |
| 33 | Amount-12MonthsNetCredit | Total credit amount in 12 months | Normal amount | 0.004654967355 |
| 34 | Amount-12MonthsAvgNetCredit | Average monthly credit amount in 12 months | Normal amount | 0.004596220721 |
| 35 | 4-5YearLPS | Late payment surcharge in last 4-5 years | Additional amount | 0.004532165693 |
| 36 | 3-4YearLPS | Late payment surcharge in last 3-4 years | Additional amount | 0.004304066086 |
| 37 | Above5YearLPS | Late payment surcharge before 5 years | Additional amount | 0.003930719617 |
| 38 | Amount-NetCreditYTD | Amount received in current year till date | Additional amount | 0.003590748039 |
| 39 | Last Discon Reason | Reason for last disconnection | Extra info | 0.003512856482 |
| 40 | Partner | Partner agency | Extra info | 0.003249072825 |
| 41 | Type-Premise | House hold type like house, flats, market etc. | Bill info | 0.003050005819 |
| 42 | Meter Category | Meter category | Extra info | 0.002990504483 |
| 43 | Category-Rate | Category for rate like resedential, commercial etc. | Bill info | 0.002828424755 |
| 44 | Load-Sanctioned | Allowed load | Extra info | 0.002666668370 |
| 45 | BankComm | Bank commission | Normal amount | 0.002146206356 |
| 46 | LoadConnected | Actual load | Extra info | 0.001827458080 |
| 47 | Industry Class | Type of industry | Extra info | 0.001569211472 |
| 48 | Payment Received | Payment received by customer | Additional amount | 0.001457066958 |
| 49 | Conn Phase | Connection phase | Extra info | 0.001277748626 |
| 50 | Amount-LPSWaived | Waived amount of late payment surcharge | Additional amount | 0.001175315316 |
| 51 | Amount-Adjustment | Pending bill adjustment | Additional amount | 0.000894321102 |
| 52 | Amount-Normal | Amount against consumed untis | Normal amount | 0.000798917360 |
| 53 | Units-Normal | Consumed units | Normal units | 0.000554570642 |
| 54 | Ordinary IP | Type of power supply in ordinary industry (AC) | Extra info | 0.000542179330 |
| 55 | Amount-Set Aside | Disputed amount | Additional amount | 0.000473512563 |
| 56 | DC Ordinary IP | Type of power supply in ordinary industry (DC) | Extra info | 0.000418416649 |
| 57 | Type-Consumer | Connection type | Bill info | 0.000393096677 |
| 58 | Code-Set Aside | Code for disputed amount | Additional amount | 0.000386602460 |
| 59 | Units-Adjusted | Adjusted units for previous month | Additional units | 0.000138101530 |
| 60 | Allowance-PreviousYear | Any allowance for previous year | Additional amount | 0.000136395061 |
| 61 | Units-Average | Guessed units written for any month | Normal units | 0.000084363601 |
| 62 | Amount-Average | Amount against guessed units | Normal amount | 0.000062844597 |
| 63 | Amount-Adjusted | Amount for adjusted units | Additional amount | 0.000015937667 |
| 64 | Amount-Clearing | Amount needed to clear account | Additional amount | 0.000001677576 |
| 65 | Connection-Status | State of connection like in-active, active | Extra info | 0.000000006279 |
| 66 | Type-Bill | Type of bill | Bill info | 0.000000000000 |
| 67 | Category-DC Rate | Rate category of DC | Extra info | 0.000000000000 |
| 68 | Request-Installement | Installement request | Additional amount | 0.000000000000 |
| 69 | Amount-DownPayment | Down payment amount | Additional amount | 0.000000000000 |
| 70 | Amount-OutStndDPayment | Remaining down payment | Additional amount | 0.000000000000 |
| 71 | 1YearLPS | Late payment surcharge in last 1 year | Additional amount | 0.000000000000 |

# Appendix B

# Simulation Parameters

Table B.1 contains the list of parameters of the classifiers used in the simulation.

TABLE B.1: List of Simulation Parameters

| Classifier | Simulation Parameters |
|---|---|
| BernoulliNB | alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None |
| GaussianNB | priors=None, var_smoothing=1e-09 |
| LDA | solver='svd', shrinkage=None, priors=None, n_components=None, store_covariance=False, tol=0.0001 |
| QDA | priors=None,reg_param=0.0, store_covariance= False,tol=0.0001 |
| Logistic Regression | penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='warn', max_iter=100, multi_class='warn', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None |
| SGD Classifier | loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_intercept=True, max_iter=1000, tol=0.001, shuffle=True, verbose=0, epsilon=0.1, n_jobs=None, random_state=None, learning_rate='optimal', eta0=0.0, power_t=0.5, early_stopping=False, validation_fraction=0.1, n_iter_no_change=5, class_weight=None, warm_start=False, average=False |
| Linear SVC | penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0, multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, verbose=0, random_state=None, max_iter=1000 |
| DT Classifier | criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort=False |
| MLP Classifier | hidden_layer_sizes=(100, ), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=0.0001, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-08, n_iter_no_change=10 |
| KNN | n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None |
| AdaBoost | base_estimator=None, n_estimators=50, learning_rate=1.0, algorithm='SAMME.R', random_state=None |
| Random Forest | n_estimators='warn', criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None |
| LGBoost | application='binary', is_unbalance='true',objective='binary', learning_rate=0.003, boosting_type='gbdt',metric=binary_logloss,num_leaves=10, min_data=50, max_depth=10 |
| XGBoost | base_score=0.5,booster='gbtree',colsample_bylevel=1,colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,max_depth=3, min_child_weight=1, missing=None, n_estimators=100,n_jobs=1, nthread=None, objective='binary:logistic', random_state=0,reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,silent=True, subsample=1 |
| CatBoost | learning_rate=0.047, depth=9, loss_function='Logloss' |

# Appendix C

# Pre-Processing, Training and Testing Code

Following is the Python 3.6 code of pre-processing, training and testing.

```
%Importing Libraries

import pandas as pd
import numpy as np
import pandas_ml as pdml

%Reading File
File=pd.read_csv("File.csv", low_memory=False)

File.BCM[(File['AssessedAmount']>0) &  (File['IRBDetectionAmount']>0)] = 'Theft'

%Data Munging
File.BCM.replace(['IRB (THEFT) - IBC','Theft',  'Normal', 'Assessed', 'Average',
    'Adjusted', 'ITG - Irregular Bill against Tariff Revi',
    'IRB (MC DISCREPANCY) - IBC','IBC -Wrong Tariff',
    'IBC - Assessed Bill  Revision', 'WRONG TARIFF', 'IRB (OTHER) REVISION',
    'WRONG READING/PUNCHING/POSTING/MMF/CALCU','Extra GST and/or Further GST Revision',
    'MIDDLE BILL'], [1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0], inplace=True)
File['Last DC Date']= File['Last DC Date'].fillna ('01-Jan-2001')
File['Last Payment Date']= File['Last DC Date'].fillna('01-Jan-2001')
File['Last DC Date']=pd.to_datetime (File['Last DC Date'],
    infer_datetime_format = True)
File['Last Payment Date'] = pd.to_datetime (File[ 'Last Payment Date'],
    infer_datetime_format = True)
SFile=File[['AccountContract', 'Billing Class', 'Rate Category', 'DC OIP',
    'DC Rate Category', 'OIP', 'Phase', 'Cycle Day', 'Sanctioned Load',
    'Connected Load', 'Status', 'Last DC Date', 'Last DC Reason',  'Premise Type',
    'Set Aside Amount', 'Set Aside Code', 'Installement Number', 'Installement Amount',
    'Consumer Type', 'Industry  Classification', 'Agency', 'Last Payment Date',
    'Last Payment Amount', 'Meter Make', 'Device Category', 'Last SIR Number',
    'Last SIR CreatedOn' , 'BillingMonth', 'BillType', 'NormalUnits', 'NormalAmount',
    'AverageUnits', 'AverageAmount', 'AdjustedUnits', 'AdjustedAmount', 'RegularUnits',
    'RegularAmount', 'CurrentUnits', 'CurrentAmount',  '12MonthsAvgUnits',
    '12MonthsAvgAmount', 'UnitBilledYTD',  '12MonthsUnits', '12MonthsAmount',
    'OpeningBalance', 'AmountBilled', 'LPSBilled' , 'LPSWaived', 'BankCharges',
    'Payment', 'Adjustment',  'PreviousYearAllowance',  'DownPaymentRequest',
    'DownPayment',  'ClearingAmount', 'ClosingBalance', 'OutstandingDownPayment',
    'A <=(366)', 'B (365)-(1)', 'C 0-30', 'D 31-60', 'E 61-90', 'F 91-180', 'G 181-365',
    'H 366-730', 'I 731-1095', 'J 1096-1460', 'K 1461-1825',  'L >=1826',
    'GrossBilledYTD', 'NetCreditYTD', '12MonthsGrossBilled',  '12MonthsNetCredit',
    '12MonthsAvgGrossBilled', '12MonthsAvgNetCredit',  'MonthYearSelection',
    'Assessed Amount', 'CurrentAmount', 'IRBDetectionAmount', 'BCM']]
SFile['Last DC Reason']=SFile['Last DC Reason'].replace([0,3,4],[1,2,3])
SFile['Last DC Reason']=SFile['Last DC Reason'].fillna(0)
SFile['DC Rate Category']=SFile['DC Rate Category'].fillna(0)
SFile['Rate Category']=SFile['Rate Category'].replace(['A1-R','A2-C_RET',
    'A2-C','B-2','B-1','A1-R_EM_BE','E-1_I','E-1_II','A2-C_B','B-3_TOD',
    'E-2_I',  'A2-C_B_RET', 'MIX', 'D-1','B-2_TOD'],
    [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])
SFile['Rate Category']=SFile['Rate Category'].fillna(0)
SFile['Billing Class']=SFile['Billing Class'].replace (['A1-R', 'A2-C',
    'B', 'E', 'MIX','D'], [1,2,3,4,5,6])
SFile['Billing Class']=SFile['Billing Class'].fillna(0)
SFile['DC OIP']=SFile['DC OIP'].replace(['ORD','IND'],[1,2])
SFile['DC OIP']=SFile['DC OIP'].fillna(0)
SFile['OIP']=SFile['OIP'].replace(['ORD','IND'],[1,2])
SFile['OIP']=SFile['OIP'].fillna(0)
SFile['Phase']=SFile['Phase'].replace(['SINGLE','POLY<20','POLY20TO90'],[1,2,3])
SFile['Phase']=SFile['Phase'].fillna(0)
SFile['Cycle Day']=SFile['Cycle Day'].interpolate()
SFile.dropna(subset=['BillingMonth'],how='all' ,inplace=True)
SFile['Status']=SFile['Status'].replace(['ACT','DIS','MOC'],[1,2,3])
SFile['Premise Type']=SFile['Premise Type'].replace(['HOUSE','SHOP', 'FLAT',
    'INDUSTRY', 'UNKNOWN', 'SCHOOL','TELECOM TOWER / PTCL', 'MOSQUE', 'BANK',
    'NOT AVAILABLE','MARRIAGE HALL', 'GODOWN',  'HOSPITAL - PRIVATE', 'CNG STATION',
    'PETROL PUMP', 'OFFICE - PRIVATE / LAWYERS /  SOLICITORS', 'MADARSA', 'RESTAURANT',
    'HOTEL',  'DISPENSARY / CLINIC / LABORATORY -GOVT', 'FACTORY', 'SOFTWARE HOUSE',
    'SHOPS','DISPENSARY / CLINIC / LABORATORY- PVT', 'OFFICE - GOVT',
    'CHARITABLE INSTITUTE / NGO / WELFARE', 'MOBILE TOWER',
    'TUBE WELL - FISH FARM, NURSERIES, FISH H', 'IMAM BARGAH', 'PARKS / PLAYGROUND',
    'HOSPITAL - GOVT', 'GRAVEYARD', 'NEON', 'OFFICE', 'STREET LIGHT', 'GRID STATION',
    'WATER PUMP', 'OFFICES', 'HOSPITALS / DISPENSERIES','COLLEGE', 'POST OFFICE',
    'FISH HATCHERIES'], [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,
    22,23,24,25,26,27, 28, 29, 30,31, 32, 33,34, 35, 36, 37, 38,39,40,41,42])
SFile['Premise Type']=SFile['Premise Type'].fillna(0)
SFile['Set Aside Amount']=SFile['Set Aside Amount'].fillna(0)
SFile['Set Aside Code']=SFile['Set Aside Code'].fillna(0)
SFile['Installement Number']=SFile['Installement Number'].fillna(0)
SFile['Installement Amount']=SFile['Installement Amount'].fillna(0)
SFile['Consumer Type']=SFile['Consumer Type'].replace(['DOL Connection',
    'DOL Connection, 3 phase (mostly)', 'Bulk LT Connection','HT Connection'],
    [1,2,3,4])
SFile['Consumer Type']=SFile['Consumer Type'].fillna(0)
SFile['Industry Classification']=SFile['Industry Classification'].replace(
    ['Small Industry', 'Low Tension  Large Industry','High Tension  Large Industry'],
    [1,2,3])
SFile['Industry Classification']=SFile['Industry Classification'].fillna(0)
SFile['Agency']=SFile['Agency'].replace(['EVALUATION GRID PRIVATE LIMITED'],[1])
SFile['Agency']=SFile['Agency'].fillna(0)
SFile['Last Payment Amount']=SFile['Last Payment Amount'].fillna(0)
SFile['Meter Make']=SFile['Meter Make'].replace(['SBL', 'PEL','CHA', 'EAC',
    'EPL','DKB','KRZ', 'SPC', 'USR','TEC', 'KOR','DMT', 'GNZ','DTM', 'DEP',
    'EEC','CAH','KBK', 'INT','MTI','ISK', 'Unknown', 'IND', 'DSB', 'LAG',
    'ACT','PAF', 'SSW','EMC','CEL', 'POL','HTL','PRI', 'SRH','SBS', 'DPE',
    'SBE','UHR', 'ABB','MPV', 'BEM','LG','VER', 'SPT','AIS', 'BIC','CRE',
    'DIN', 'ELS'],[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,
    25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49])
SFile['Meter Make']=SFile['Meter Make'].fillna(0)
SFile['Device Category']= SFile['Device Category'].replace (
    ['METER; ENERGY SINGLE PHASE 10-40A', 'METER; STATIC SINGLE PHASE',
    'METER; ENERGY 3 PHASE 15-90A', 'METER; STATIC THREE PHASE',
    'METER; 3PH DOL AMR GPRS/3G W/DC SW',
    'METER; HT CT/PT&LT OPERATED W/GPRS MODEM', 'METER; STATIC LT C.T.O PROG.',
    'METER; ENERGY SINGLE PHASE (REFURBISHED)', 'METER; 1PH DOL AMR GPRS/3G W/DC SW',
    'METER; ENERGY C.T OPERATED 100/5A', 'METER; ENERGY THREE PHASE (REFURBISHED)',
    'METER;DIRECT ONLINE W/GPRS MODEM 3 PHASE', 'METER E 42 F-D/M 400/5A,66000/100 V',
    'USED METER E 42 F-DM 400/5A, 66000/100 V', 'USED METER; STATIC LT C.T.O PROG.'],
    [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])
SFile['Device Category']=SFile['Device Category'].fillna(0)
SFile['Last SIR Number']=SFile['Last SIR Number'].fillna(0)
SFile['Last SIR CreatedOn']=SFile['Last SIR CreatedOn'].fillna(0)
SFile['BillType']=SFile['BillType'].replace(['Regular','IRB-Detection',
    'IRB-Revised'],[1,2,3])
SFile['PreviousYearAllowance']=SFile['PreviousYearAllowance'].fillna(0)
SFile['DownPaymentRequest']=SFile['DownPaymentRequest'].fillna(0)
SFile['ClearingAmount']=SFile['ClearingAmount'].fillna(0)
SFile['ClosingBalance']=SFile['ClosingBalance'].fillna(0)
SFile['OutstandingDownPayment']=SFile['OutstandingDownPayment'].fillna(0)
SFile['A <=(366)']=SFile['A <=(366)'].fillna(0)
SFile['B (365)-(1)']=SFile['B (365)-(1)'].fillna(0)
SFile['C 0-30']=SFile['C 0-30'].fillna(0)
SFile['D 31-60']=SFile['D 31-60'].fillna(0)
SFile['E 61-90']=SFile['E 61-90'].fillna(0)
SFile['F 91-180']=SFile['F 91-180'].fillna(0)
SFile['G 181-365']=SFile['G 181-365'].fillna(0)
SFile['H 366-730']=SFile['H 366-730'].fillna(0)
SFile['I 731-1095']=SFile['I 731-1095'].fillna(0)
SFile['J 1096-1460']=SFile['J 1096-1460'].fillna(0)
SFile['K 1461-1825']=SFile['K 1461-1825'].fillna(0)
SFile['L >=1826']=SFile['L >=1826'].fillna(0)
SFile['GrossBilledYTD']=SFile['GrossBilledYTD'].fillna(0)
SFile['NetCreditYTD']=SFile['NetCreditYTD'].fillna(0)
SFile['12MonthsGrossBilled']=SFile['12MonthsGrossBilled'].fillna(0)
SFile['12MonthsNetCredit']=SFile['12MonthsNetCredit'].fillna(0)
SFile['12MonthsAvgGrossBilled']=SFile['12MonthsAvgGrossBilled'].fillna(0)
SFile['12MonthsAvgNetCredit']=SFile['12MonthsAvgNetCredit'].fillna(0)
SFile['Set Aside Code']=SFile['Set Aside Code'].replace(
    ['Y','K','J','G','8','7','4'],[1,2,3,5,8,7,4])
SFile['OpeningBalance']=SFile['OpeningBalance'].fillna(0)
SFile['AmountBilled']=SFile['AmountBilled'].fillna(0)
SFile['LPSBilled']=SFile['LPSBilled'].fillna(0)
SFile['LPSWaived']=SFile['LPSWaived'].fillna(0)
SFile['BankCharges']=SFile['BankCharges'].fillna(0)
SFile['Payment']=SFile['Payment'].fillna(0)
SFile['Adjustment']=SFile['Adjustment'].fillna(0)
SFile['DownPayment']=SFile['DownPayment'].fillna(0)

%Train Test Split:
from sklearn.model_selection import train_test_split
```

```
train, test = train_test_split(SFile, test_size=0.2)

train['BCM'].value_counts(dropna=False)
test['BCM'].value_counts(dropna=False)
train.to_csv('i.csv', index=False)
train_set['BCM'].value_counts(dropna=False)
test_set['BCM'].value_counts(dropna=False)
train_set.to_csv('F:\\train_data.csv', index=False)
test_set.to_csv('F:\\test_data.csv', index=False)

%Training and testing of CatBoost Classifier:

from catboost import CatBoostClassifier

TrainFile=pd.read_csv("F:\\train_data.csv", low_memory=False)
Theft_x=TrainFile[[all feature names]]
Theft_y=TrainFile.BCM
CB_Theft_Model = CatBoostClassifier(learning_rate=0.047, depth=9,
    loss_function='Logloss')
CB_Theft_Model.fit(Theft_x,Theft_y,cat_features=[])
TestFile = pd.read_csv('F:\\Phd\Data\\Assessed Data\\test_data.csv',
    low_memory=False)
Theft_x_test=TestFile[[all feature names]]
Theft_y_test=TestFile.BCM
Theft_CB_cat=CB_Theft_Model.predict(Theft_x_test)

%Building Accuracy Matrix

CB_cat_Theft_accuracy=(metrics.accuracy_score(Theft_y_test,Theft_CB_cat)*100)
CB_cat_conf_theft=metrics.confusion_matrix(Theft_y_test, Theft_CB_cat)
CB_cat_tp_theft=CB_cat_conf_theft[1,1]
CB_cat_tn_theft=CB_cat_conf_theft[0,0]
CB_cat_fp_theft=CB_cat_conf_theft[0,1]
CB_cat_fn_theft=CB_cat_conf_theft[1,0]

print('CB True Positive Theft=',CB_cat_tp_theft)
print('CB True Negative Theft=',CB_cat_tn_theft)
print('CB False Positive Theft=',CB_cat_fp_theft)
print('CB False Negative  Theft=',CB_cat_fn_theft)

CB_cat_percision_theft=CB_cat_tp_theft/(CB_cat_tp_theft+CB_cat_fp_theft)*100
CB_cat_recall_theft=CB_cat_tp_theft/(CB_cat_tp_theft+CB_cat_fn_theft)*100
CB_cat_fmeasure_theft=2 * CB_cat_percision_theft * CB_cat_recall_theft /
    (CB_cat_percision_theft + CB_cat_recall_theft)
CB_cat_accuracy_theft=(CB_cat_tp_theft + CB_cat_tn_theft) /
    (CB_cat_tp_theft + CB_cat_tn_theft + CB_cat_fp_theft + CB_cat_fn_theft)

print("CB Percision for thieves : "+str(CB_cat_percision_theft)[:5]+"%")
print("CB Recall for thieves : "+str(CB_cat_recall_theft)[:5]+"%")
print("CB FMeasure for thieves : "+str(CB_cat_fmeasure_theft)[:5]+"%")
print("CB Accuracy for thieves : "+str(CB_cat_accuracy_theft)[:5])
```

# Appendix D

# Publication List

## D.1 Journal Articles

Following is the publication list of International Journal papers:

- **Ghori, KM.**, Abbasi, RA., Awais, M., Imran M., Ullah, A., Szathmary, L. (2019). "Performance analysis of different types of machine learning classifiers for non-technical loss detection". In: *IEEE Access*, pp. 16033–16048. **Impact Factor: 3.745 (2019)**.
- **Ghori, KM.**, Imran, M., Nawaz, A., Abbasi, RA., Ullah, A., Szathmary, L. (2020). "Performance analysis of machine learning classifiers for non- technical loss detection". In: *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16. **Impact Factor: 4.594 (2019)**.

## D.2 Conference Proceedings

Following is the conference proceeding:

- **Ghori, KM.**, Abbasi, RA., Awais, M., Imran, M., Ullah, A., Szathmary, L. (2020). "Impact of Feature Selection on Non-technical Loss Detection". In: *2020 6th Conference on Data Science and Machine Learning Applications (CDMA)*, pp. 19–24.

## D.3 Other Publications

- Rahman, S., Irfan, M., Mohsin, M., **Ghori, KM.**, Shumayla, Y., Awais, M. (2020). "Performance analysis of boosting classifiers in recognizing activities of daily living". In: *International Journal of Environmental Research and Public Health*, pp. 1082. **Impact Factor: 2.849 (2019)**.