4. UNMETERED POWER LOSSES ANALYSIS FRAMEWORK

4.1 Introduction

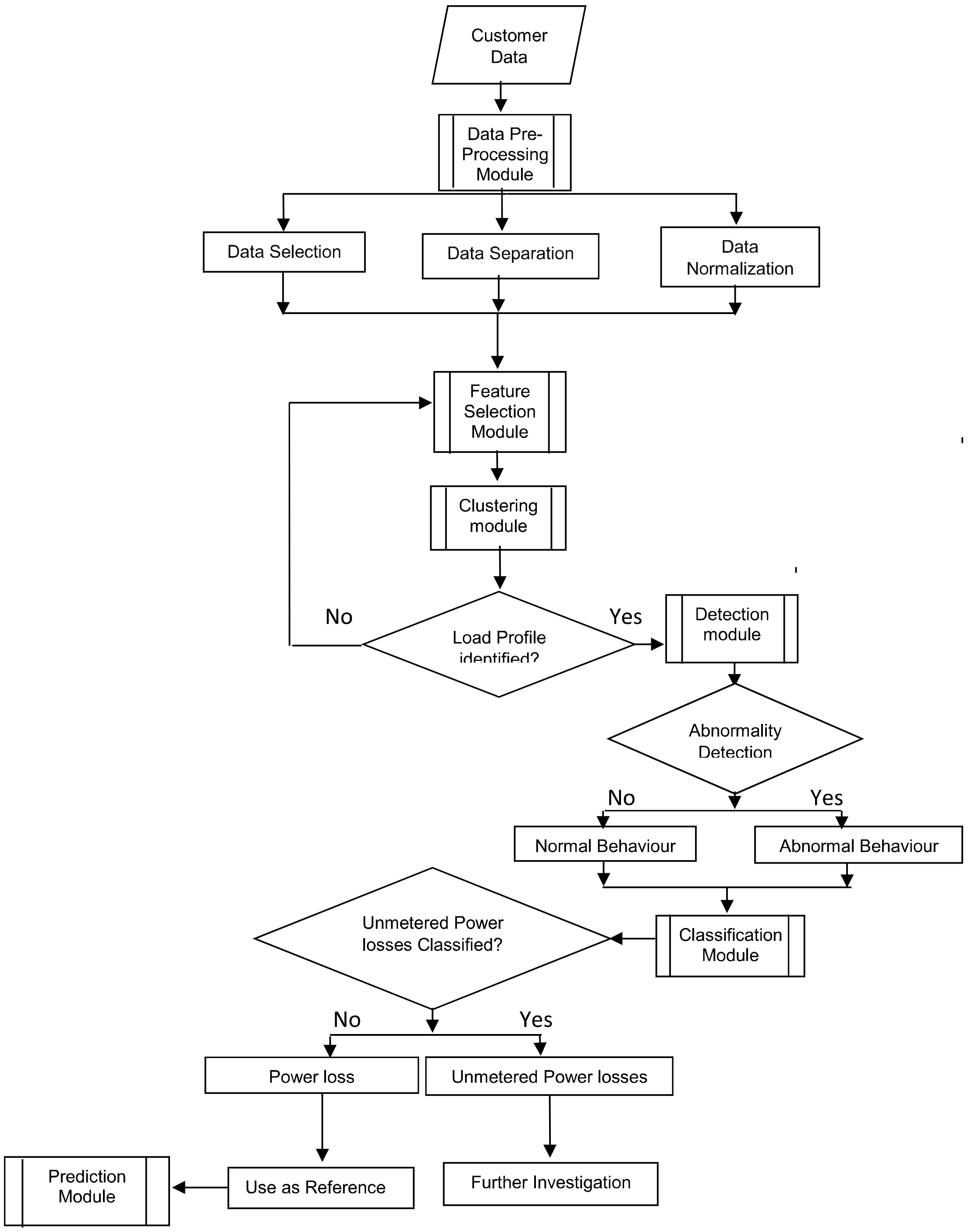
The present study to make the focus on MGVCL, Gujarat, India, Gujarat's moderate power distribution company, focus on load profiles as related of consumer behaviour. The major objective is that to expand the work refer above into a study of abnormal load behaviour as a means of finding discontinuities where these can be related to unmetered power losses. The process of study is comparing the efficiency of the Support Vector Machine [13] algorithm with the new algorithm of Extreme Learning Machine [9] and its Online Sequential-Extreme Learning Machine [10] differ as means of classification and decision in this situation. As cited in chapter-2 the literature on data pre-processing, this will makes it clear that it will improve data mining quality and to complete the parameter of the process task is very important. The main proposed objective of the unmetered power losses analysis framework that the normal behaviour of every consumer based on collected data of historical is related to find the detection and identification of discontinuity. As shown in Figure 4.1, the first step is to develop filter for pre-processing of data to obtain good quality data. These process includes normalization and to take care of missing data. In the subsequent sections, the database utilization has been described for all the subsequent experiments and brief of the proposed analysis framework is demonstrated.

4.2 The proposed unmetered power losses analysis framework

- Representative load profiles are acquired from the load profiling module as a reference. In the present study, these untypical load profiles are used as benchmarks for investigation by means of outlier detection techniques [57]. While in oldest researches such as most study have rejected classified load profiles because of undesirable behaviour.
- Based on load profiles, types of benchmarks are pattern for an abnormal behaviour s and patterns for a normal behaviour which are used as training data. The patterns for abnormal are to be examined further.
- For detecting unmetered power losses activities, provide reference points for typical load profiles when they are compared with newly emerging load profiles. If no outliers are apparent, then the new load profiles are updated. But if there is some significant deviation, then the new load profiles concerned are investigated using one of the alternative statistical-based, density-based, distance-based, model-based, or deviation-based outlier detection techniques.
- If outlier detection is confirmed as unmetered power losses activity, then the new load profile is updated as an anomalous reference for forecasting purposes. This load profile is used as a reference for forecasting indications of unmetered power losses activities among new load profiles by applying forecasting techniques such as support vector machine and time-series analysis. These forecasting techniques are to be explored later below.

- The sequence of processes involved in identifying, detecting, and predicting unmetered power losses by means of analysing the customer load consumption database of a power utility, with reference to Gujarat's MGVCL as shown in Figure 4-1. The main tasks are summarized above in this unmetered power losses analysis framework.

Figure 4.1 Proposed Framework Analysis for power losses



4.3 Data Pre-processing

In pre-processing of electricity customer [15] databases the major problem is missing or incomplete data and consistent as well as noisy data. The reasons for bad data appearing in the database are due to the extended size of the database itself like equipment faults or meter malfunction, intention human interference and human faults. The improvement in overall data quality is equally important for efficiency/ accuracy of mining results and speed of the process. There are several data performing techniques reported in several research articles as a tool of achieving these objectives. Three of the most common, namely data integration, data transformation and data cleaning are described below.

4.3.1 Data Cleaning

Data cleaning is the solution for missing or incomplete data values and for inconsistent and noisy data. In such cases, it functions by filling in missing values, smoothing noise, and resolving inconsistencies. These issues may result from human mistakes or from equipment faults that can affect the quality of data pre-processing and thus affect the quality of the results of subsequent data mining tasks. Several studies have reported on data cleaning techniques, including the wavelet multi-resolution technique, Kohonen Self Organizing Maps, the chi-square test, and regression techniques. In load profile studies such, the data is smoothed using wavelet multi-resolution to eliminate white noise. Missing values associated with weather conditions and other factors affecting the completeness of load consumption data are replaced using an estimation method that involves inserting the mean of the surrounding values in their place.

4.3.2 Data Transformation

Normalization techniques may improve data mining results as they scale the measured values to a specific range, for example [-1, 1] or [0, 1]. In power utility customer data such as that accumulated by MGVCL, some commercial customers may have larger consumption values that can outweigh other smaller-scale consumption patterns of customers. The result is a bias in distance measures. Therefore, the customer load data requires normalization for pattern comparison purposes and this forms one of the data pre-processing stages. The normalizing factor can be either the average power consumed in a certain time period or the peak power consumption of the load profiles. However, according to the peak consumption normalizing factor has certain limitations that can give misleading results. Four normalization techniques reported in the literature include decimal scaling, min-max normalization, normalization using Z-scores, and logarithmic normalization.

4.3.3 Data Integration

Data mining tasks most often involve multiple databases and data integration can merge data from these disparate sources. The data is segregated according to the load conditions and divided into groups of similar consumers presenting similar characteristics. It is necessary to classify the data in terms of the days of the week, as different load-shape patterns for working days and weekend days. Weather records are among the important factors in load research, including information detailing ambient temperature, humidity, wind speed, sky cover, and sunshine periods. Data integration techniques are needed to merge the load demand data obtained from customer metering with weather data from the Meteorological Department.

Several datasets are used for the present experiments, including commercial customers load consumption, weather data, event data, and load indices derived from the load curves. The load profiles of each individual customer are provided in Appendices A to I below. The customers are divided into the two major supply areas of GUVNL and MGVCL, Gujarat. The daily data are captured for the years from 2013 to 2014 and subsequently separated into the two categories of training data and testing data.

The three tasks involved in this module are data selection, data separation, and data normalization and each is discussed below.

Data selection – MGVCL Gujarat data required for this study consists of the accumulated records of 251 commercial customers from 2013 to 2014.

Data separation – separation of the customer data is based on types of days of the week from Monday until Sunday with additional citations for public holidays. The need to establish the relevant load conditions is crucial because it has been found in previous studies that differences in these conditions yield varying load condition for every consumers. Therefore, for the purposes of this study, it was determined to cultivate a sample of illustrative load profiles for commercial customers under the load conditions category of "type of seasons". The pertinent empirical details are set out in the several ensuing tables and figures. We are divided all customer according to season of the year. There three types of customer's monsoon, summer and winter as shown in figure-4.2, figure-4.3 and figure-4.4 respectively.

Table 4.1 Number of customers provided by MGVCL, Gujarat based on the two city selected

City	Area	No. of Customer	Mean	Std. Deviation
BARODA O&M CIRCLE	HT Exp	10	102.89	9.078
	Industrial	31	104.09	8.890
	GIDC	07	106.45	8.666
	Urban	23	104.10	7.455
	Jyotigram	28	105.39	7.219
	Agriculture	31	103.11	4.900
ANAND O&M CIRCLE	HT Exp	22	102.67	4.789
	Industrial	26	104.33	4.611
	GIDC	25	105.75	7.593
	Urban	13	102.00	7.566
	Jyotigram	18	103.05	5.785
	Agriculture	17	102.91	5.075
Total		251	103.895	6.802

Table 4.2 Load conditions

Load conditions	Items
Type of customer	Domestic
	Commercial
	Industrial
Location	Urban
	Rural
Voltage Level	Low Voltage
	High Voltage
Type of climate	Cold
	Rainy
	Hot
Weather Conditions	Visibility
	Wind Speed
	Sea Level Pressure
	Temperature
	Humidity
Type of day	Summer
	Winter
	Monsoon

The immediate task is to separate the available data into smaller datasets according to the Load conditions listed in Table 4-2.

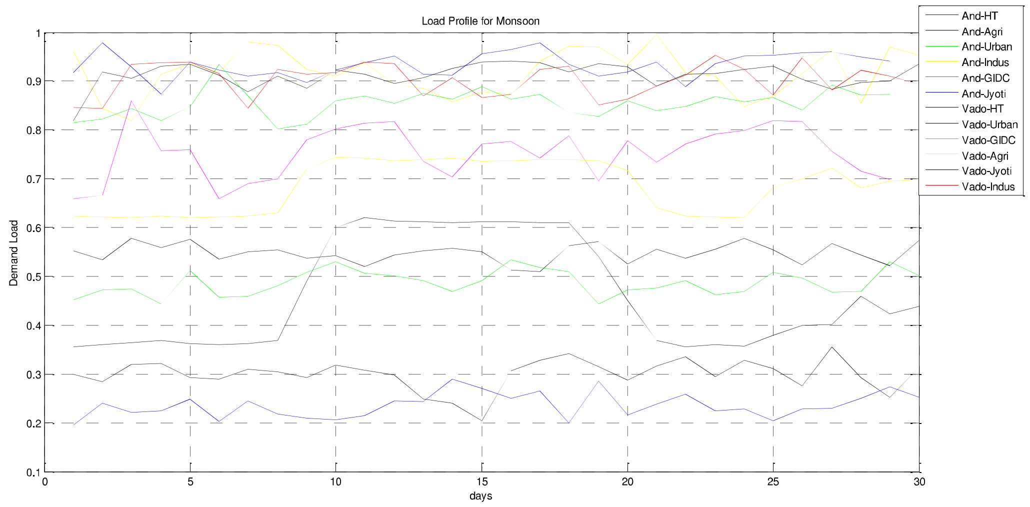


Fig-4.2 Customers' load profiles of monsoon

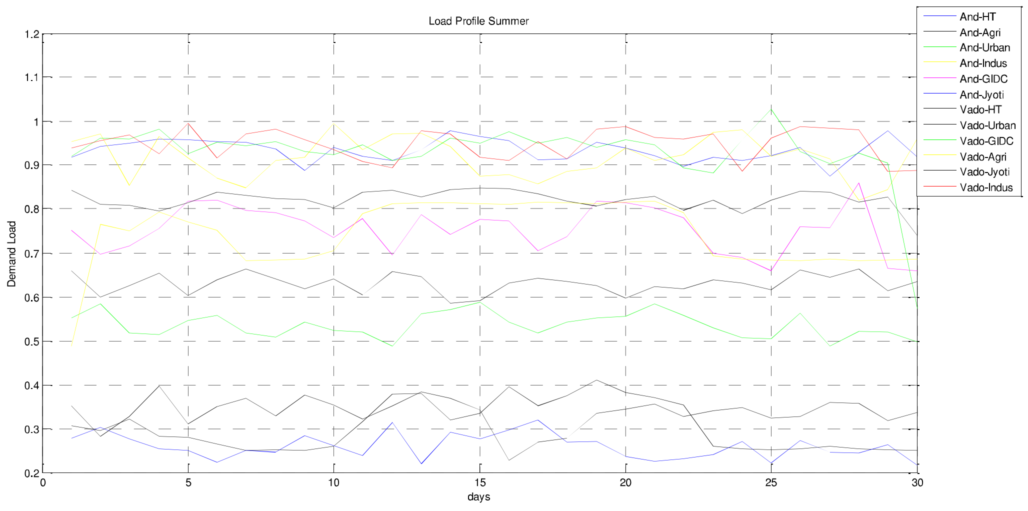


Fig-4.3 Customers' load profiles of summer

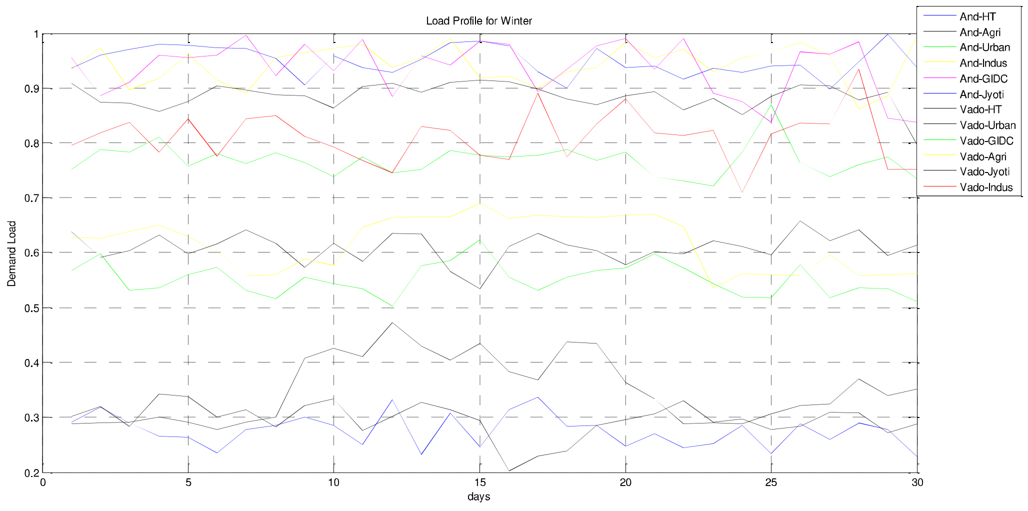


Fig-4.4 Customers' load profiles of winter

4.3.4 Data normalization

The load demand data is normalized into a common scale of [0, 1] to avoid higher weights for the peak hours. The formula selected for this purpose here is:

$$v'(i) = \frac{v(i) - \min(v(i))}{\max(v(i)) - \min(v(i))}$$

Where v(i) is the load, min v(i) is the minimum value of the given load for a given period, max v(i) is the maximum value of the given load for a given period.

Real-world data sets tend to be incomplete, noisy, and inconsistent. This applies to the utility customer data to be examined here. Data cleaning tasks attempt to fill in missing values and smooth out noise, while also identifying outliers and correcting inconsistencies. In determining a typical load profile, the three important preparatory steps to be taken are data smoothing, data normalization, and data separation.

Each load profile is characterized by a vector for a selected group of n customers,

$x^n = \{x_m{}^{(n)}\ m = 1, \dots \dots \dots, M\}$, Whose M components can be the load profile in time domain, each of 30 minutes interval, the whole set of data is then designated as X= $\{x^{(n)}, n = 1, \dots \dots \dots, N\}$.

Most previous research of relevance here involved conducting field measurement studies to collect and select the data. In some cases, stratified sampling was used, while in others, for example in one Vadodara study, comprehensive load surveys were undertaken. MGVCL Gujarat began its Automatic Meter Reading (AMR) (For selected consumers and only research purpose) project in 2013 and it was this data that was collected from the MGVCL Gujarat, Metering Department for the present research purposes. In addition to the load consumption data, weather data and commercial indices have been acquired to enable any correlations between these factors and customer loads to be considered [29].

The first step in data pre-processing is to extract and clean the data from the text files provided by MGVCL. One particular commercial customer can have two meters, a main meter and a check meter. These meters complement each other in that if the main meter is not functioning, the reading from the check meter is used as an alternative. The metering attributes assessed here are ID, date, time, kWh-in, kWh-out, kVar-in, and kVar-out. In this study, only the kWh data is used. In order to compare and obtain the final reading, the following source code has been developed. With most commercial customers being supplied with both a main meter and a check meter, the algorithm to acquire meter readings for the purpose of data pre-processing from both meters is set out as shows in fig-4.5.

The electricity customer consumption data gathered from the electronic metering were normalized because this data has been needed to represent with a common scale for comparison purposes. In the present study, the data has been normalized in the specific scale of [0, 1] by using as the normalizing ratio the peak value of the pattern over the time interval of the definition.

$$\mathrm{NL} = \frac{\mathrm{X} - \mathrm{Min(X)}}{\mathrm{Max(X)} - \mathrm{Min(X)}}$$

All the individual normalized customer load demands used for comparisons here are detailed in fig-4.2. In these Appendices, the figures show the load profiles for 251 customers that have been separated based on types of days comprising weekdays from Monday to Friday, Saturdays, Sundays, and public holidays. The total weekday profiles from 2013 to 2014 and the total yearly load profiles for the same period have also been generated for each individual customer. The data used for this study consists of 70 attributes gathered from different data sources. The customer data that allows for time factors, weather data, and events data were provided by the two types of metering, main reading and check reading.

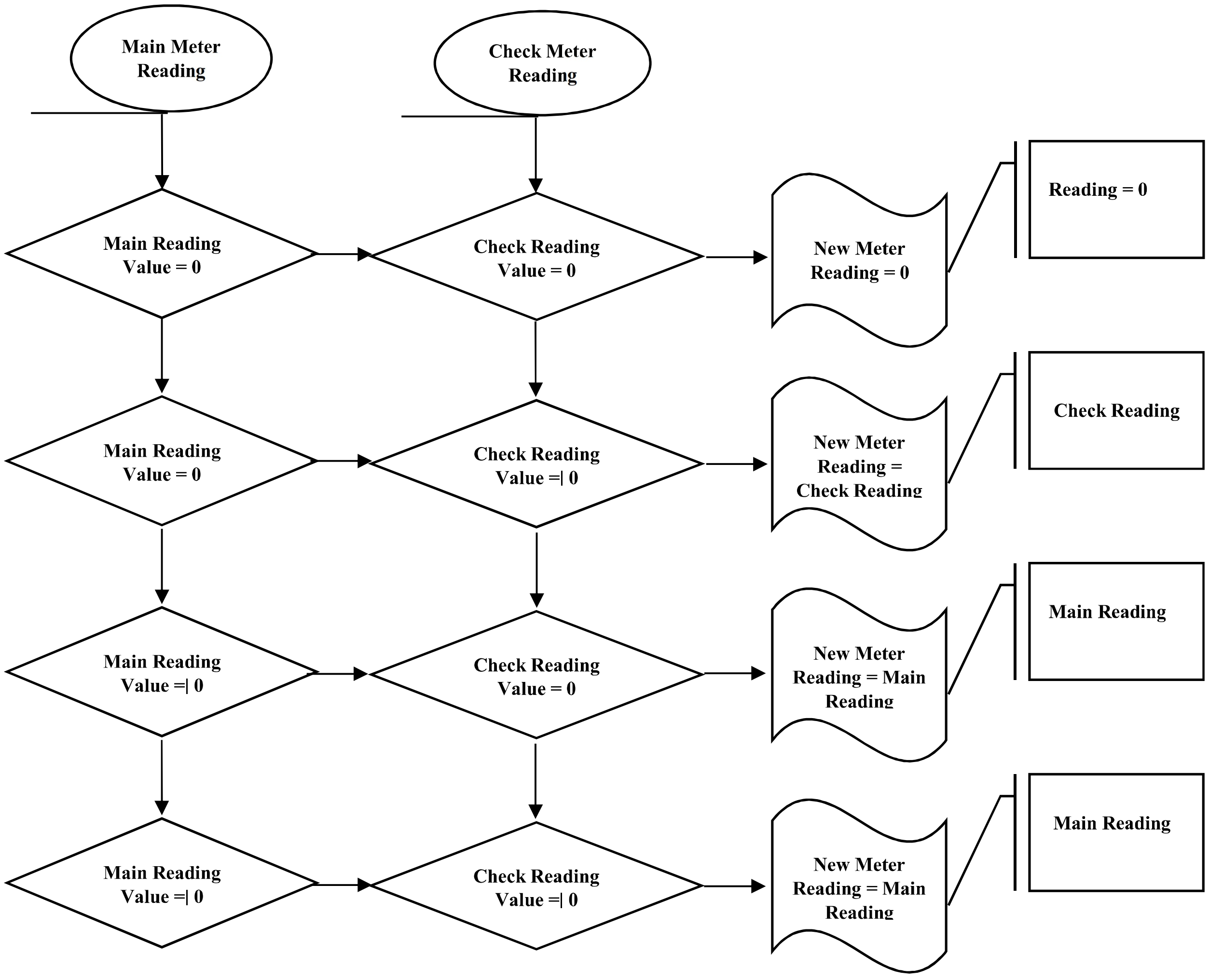


Fig-4.5 Dealing with new values based on data from main meter reading and check meter reading

4.4 Detection

For the labelling of the pre-defined data, the detection [14] process is designed. In the procedure of load pattern naming are used three definitions. The detected unmetered power losses known to be analysed as zero consumption will be defined named as confirmed. There are two techniques for detection as study in the reviewed literature. These are deviation-based outlier and statistical outlier detection which have been take in the proposed research. There are many criteria have been produced through the bunch task in module 3. From the bunch generated, we initialize these criteria as list out below [17].

- Case-1- If the power consumption is zero from 1 to 30 days, then it is absolutely an unmetered power losses due to faulty metering.

- Case-2 - If the power consumption within the range generated by the clustering techniques, then the load power consumption is fall in normal behaviour on that particular day.
- Case-3 - If the power consumption detect outside of the range generated by the clustering techniques, then the load power consumption is fall in suspicious behaviour on that particular day. Its required on-site field investigation and it is generate an alert to maintain department for observation.

The detection steps design to investigate the class name for the ensuing classification task are as follows.

Case 1: To check for any zero consumption

```
Sub Case 1: (Set the Class: Unmetered Power Losses or Power Losses)
Definition of variables and attributes.
Set attributes and variables.
Select the respective customers.
For each Load Profile (rows):
If any value from $X_t$= {$X_1, X_2$ , ............ $X_{30}$} is equal to zero, t is the one day time interval
Then the class value = “Unmetered Power losses”
Else the class value = “Power Losses”
End If
Next Load Profile
End Sub Case 1
```

Case 2: Set the parameters

```
Sub Case 2: Setting up the parameters
Group the consumers based on the seasons of the year: Summer, winter, Monsoon
Set the Mean(M) for every group.
Set Standard Deviation(SD) for every group.
Set Upper Limit(UL), $L= \mu + 3\sigma$
Set Lower Limit(LL), $L =\mu+ 3\sigma$
End Sub Case 2
```

Case 3: To check abnormalities in unmetered power losses

```
Sub Case 3 (Check Unmetered Power Losses: Abnormal or Normal)
 Attributes and variables declaration.
Initialization variables and attributes.
Carefully chosen unmetered power losses datasets calculated from Sub Case1.
For each Unmetered Power Losses Load Profile (rows):
If the case value = “Unmetered Power Losses” and value.lower $\leq x_i \geq$ value.upper
Then the unmetered power losses case value = “Normal Behaviour”
Else the unmetered power losses case value = “Suspicious Behaviour”
End If
Next Unmetered Power Losses Load Profile
End Sub Case 3
```

4.5 Customer Load Analysis

Time factors, including the day of the week and the hour of the day, are very important as contributing factors in determining load changes. This is because load demands can vary in accordance with different hours, days, weeks, and months. Such demands can behave differently on different weekdays, as between weekdays and weekends, as between seasons, and during public holidays and the adjacent days.

4.6 Weather Factor

Various weather variables, including temperature, humidity, wind speed, and sky cover, have been considered in several load profile studies for many countries, Temperature and humidity are the most commonly used variables. However, the nature of the impact of these variables on electricity supply system loads in India is yet to be discovered and confirmed. Although Indian weather is known to be seasonal throughout the year, it is nonetheless important to conduct an experiment to investigate whether commercial customers in this country may have some varying responses to such factors. Therefore, Indian weather data, particularly for Gujarat, was downloaded from the weather Department online resource. The particular information obtained consisted of temperature and relative humidity details from 1 January 2012 to 31 December 2014.

4.7 Key Algorithms for Customer Behaviour Classification and Prediction

The three algorithms Extreme Learning Machine, OS-Extreme Learning Machine, and support vector machine have been selected for use in the classification and prediction procedures that are applied to electricity utility customer behaviour in the present context. All these algorithms additionally apply both the sigmoid and Radial Basis Function nodes activation functions.

4.7.4 Extreme Learning Machine

Extreme Learning Machine [9] was proposed by Huang for single hidden-layer feed-forward neural networks (SLFNs) and it was so devised as to produce superior performance. This algorithm was claimed to be extremely fast in its learning speed and to have better generalization performance when compared to conventional algorithms. Learning speeds that are claimed to be slower due to a slow gradient-based iterative learning algorithm are used extensively to train neural networks. Unlike many other popular learning algorithms, little human involvement is required in Extreme Learning Machine. Except for the numbers of the hidden neurons (insensitive to Extreme Learning Machine), no other parameters need to be tuned manually by users because this algorithm chooses the input weights randomly and analytically determines the output weights.

Usually, feed-forward neural networks of a single hidden-layer has three kinds of input parameters: the input weight w_i, the hidden neuron biases b_i, and the output weight β_i. While, conventional learning algorithms of feed-forward neural networks of a single hidden-layer have to tune these three types of parameters, Extreme Learning Machine just randomly generates the input weight w_i and the hidden neuron biasesb_i, and then analytically calculates the output weight β_i. No further learning is required for Single Hidden-Layer Feed-Forward Neural Networks trained by Extreme Learning Machine. The Extreme Learning Machine is a general learning algorithm for Single Hidden-Layer Feed-Forward Neural Networks that works effectively for function approximations, classifications, and online prediction problems. Moreover, it can generally work well for a variety of types of applications. Given N arbitrary distinct samples (X_t, T_t),

$where\ X_t = [X_{t1}, X_{t2}, \ldots\ldots, X_{tn}]^T \epsilon R^n\ and \qquad T_t = [T_{t1}, T_{t2}, \ldots\ldots, T_{tn}]^T \epsilon R^m$,
And, Single Hidden-Layer Feed-Forward Neural Network with N hidden neurons and activation function g(x) can be mathematically modelled as

$$\sum_{i=1}^{N} \beta_i g\ (w_i x_j + b_i) = O_j, j = 1, \ldots\ldots, n \qquad \ldots\ldots\ldots\ldots\ldots\ldots (4.1)$$

Where w_i is the weight vector connecting the input neurons and the i^{th} hidden neuron, β_i is the weight vector connecting the i^{th} hidden neuron and output neurons, and b_i is the threshold of the i^{th} hidden neuron. Here $w_i x_j$ denotes the inner product of w_i & x_j if the Single Hidden-Layer Feed-Forward Neural Networks can approximate these N sample with zero error, then $\sum_{j=1}^{n} || O_j - t_j || = 0$ follows; i.e. there exist β_i, w_i, b_i such that $that$ $\sum_{i=1}^{N} \beta_i\ g\left(w_i . x_j, + b_i\right) = t_j, j = 1, \ldots\ldots, n$. The above n equation can be written compactly as HB=T, where H $(w_1, \ldots, w_N, b_1, \ldots, b_N, x_1, \ldots, x_N) =$

$$\begin{bmatrix} g(w_1 . x_1 + b_1) & \ldots\ldots & g(w_N . x_1 + b_N) \\ \vdots & \ldots\ldots & \vdots \\ g(w_1 . x_n + b_1) & \ldots\ldots & g(w_N . x_n + b_N) \end{bmatrix}_{N \times n} \qquad \ldots\ldots\ldots\ldots\ldots (4.2)$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_N^T \end{bmatrix}_{N \times m} \text{ and } T= \begin{bmatrix} t_1^T \\ \vdots \\ t_n^T \end{bmatrix}_{n \times m} \qquad \ldots\ldots\ldots\ \ldots\ldots (4.3)$$

As specified in Huang and Babri [109], *H* is called the hidden-layer output matrix of the neural network, with the i^{th} column of *H* being the i^{th} hidden neuron output with respect to inputs$x_1, x_2, \ldots\ldots, x_N$. Based on their previous work, matrix *H* is square and invertible only if the number of hidden neurons is equal to the number of distinct training samples, *N* = *n* , indicating, therefore, that feed-forward neural networks of a single hidden-layer can approximate these training samples with zero error. In most cases, the number of hidden neurons is much lower than the number of distinct training samples, *N* << *n*, *H* is a non-square matrix and there may not exist $w_i, b_i \beta_i (i = 1, \ldots, n)$ such that Hβ = T. Thus, one specific set of $\hat{w}_i$,$\hat{b}_i \hat{\beta}_i$(i=1,…,N) needs to be found such that $||\mathrm{H}(\hat{w}_i, \ldots, \hat{w}_N, \hat{b}_1, \ldots, \hat{b}_N)\ \hat{\beta} - T\ || = \min_{w_i b_i \beta} ||\mathrm{H}(w_1, \ldots, w_N, b_1, \ldots b_N)\beta - T||$

Which is equivalent to minimizing the cost function $E = \sum_{j=1}^{n} \left(\sum_{j=1}^{N} B_i\ g\left(w_i . x_j + b_i\right) - t_i{}^2 \right)$
As Huang maintains, the hidden neuron parameters need not be tuned as the matrix *H* indeed converts the data from nonlinear separable cases to high dimensional linear separable cases. Simulations of numerous real-world datasets (with noise) have shown that Extreme Learning Machine performs well in such different cases. However, Huang showed that the input weights and hidden neurons or kernel parameters are not necessarily tuned and can be randomly selected and then fixed. In fact, the parameters of these hidden neurons are not only independent of each other, but also independent of the training data. Thus, for fixed input weights and the hidden layer biases or kernel parameters, training feed-forward neural

networks of a single hidden-layer is simply equivalent to finding a least squares solution $\hat{b}$ for the linear system = T . The unique smallest-norm least squares solution for the above linear system is $\hat{b} = H^{\dagger}T,$ where $H^{\dagger}$ is the Moore-Penrose generalized inverse of the hidden-layer output matrix H. As confirmed by Huang [109], this method may tend to achieve good generalization performance. The Extreme Learning Machine algorithm consisting of only three steps can be summarized as set out below. For a given a training set $\aleph = \{x_i, t_i)|x_i \epsilon\ R^n, i = 1, \ldots., n\}$, activation function g(x), and hidden neuron number n.
step1: Assign random imput weight w^i and bias $b_i, i = 1, \ldots., N$.
step2: Calculete the hidden- layer output matrix H.
step3: Calulete the output weight β: β=$H^{\dagger}$T. where H,β and T are defined as formula (2) and (3).
Extreme Learning Machine can accommodate the nonlinear activation function and kernel function. Furthermore, it can avoid difficulties such as stopping criteria, learning rate, learning epochs, and local minima very effectively and in this respect, it is unlike other tuning or adjustment methods.
Huang's analyses established that there are several methods to calculate the Moore-Penrose generalized inverse of a real or complex matrix. These methods may include, but are not limited to, orthogonal projection, orthogonalization method, iterative method, and Singular Value Decomposition (SVD). The orthogonalization method and iterative method have their limitations because searching and iteration are used and these are processes that should be avoided in Extreme Learning Machine applications. The orthogonal projection method can be used when H*H is non-singular and $H^{\dagger}$ =$(H * H)$ ^{-1}H* = - . However, H*H may not always be non-singular or may tend to be singular only in some applications and, as a consequence, the orthogonal projection method may not perform well in all applications. It is, then, the SVD that can be generally be used to calculate the Moore-Penrose generalized inverse of H in all cases. Extreme Learning Machine does not require the matrix H to be invertible as the SVD method is used in the boosting phase. Moreover, it remains suitable for non-invertible matrices. Extreme Learning Machine provides for a very fast learning capability compared with the alternatives. In, the performance of the Extreme Learning Machine algorithm was compared with other algorithms, including suppot vector machine and conventional back-propagation. The results showed that Extreme Learning Machine has an outstanding performance in relative terms. Furthermore, as Huang and Chen [110] have rece Power loss proven, Extreme Learning Machine is actually a Learning algorithm for generalized single hidden-layer feed-forward networks. Besides sigmoid Networks and radial basis function networks, such Single Hidden-Layer Feed-Forward Neural Networks also include trigonometric networks, threshold networks, fuzzy inference systems, fully complex neural networks, high-order networks, ridge polynomial networks, and wavelet networks.

4.7.5 Online Sequential Extreme Learning Machine

In order to handle online applications, the variant of Extreme Learning Machine referred as Online Sequential-Extreme Learning Machine [10] has been proposed. This algorithm was originally developed for feed-forward networks of a single hidden-layer with radial basis function or additive hidden nodes in a unified proposed analysis. The Online Sequential-Extreme Learning Machine can learn data chunk-by-chunk and that accommodates its application in real-world Industrial contexts. According to, Online Sequential-Extreme

Learning Machine also works for other types of feed-forward neural networks of a single hidden-layer that may not be neural networks.

In the derivation of sequential Extreme Learning Machine, only the specific matrix H is considered, where the rank of H is equal to the number of hidden neurons: *rank (H)=N*. Under this condition, the following implementation of the pseudo inverse of H is readily derived and given by $H^{\dagger}$=$(H^T \quad H)^{-1}$ $H^{\dagger}$, which is often called the left pseudo-inverse of H from the fact that $H^{\dagger}$H=I_N. The corresponding estimation of β is given by $\hat{\beta}$= $(H^T \quad H)^{-1}$ H^TT which is called the least squares solution to *H*β = *T*.

In, Liang [90] reviewed the Extreme Learning Machine and Online Sequential-Extreme Learning Machine approaches. Online Sequential-Extreme Learning Machine consists of two main phases, specifically the boosting phase or initialization phase and the sequential-learning phase. The boosting phase trains the Single Hidden-Layer Feed-Forward Neural Networks using the primitive Extreme Learning Machine method given some batch of training data in the initialization stage. These data are discarded once the process is completed. After that, the Online Sequential-Extreme Learning Machine will learn the training data chunk-by-chunk. Subsequent Power loss, all the training data will be discarded once the learning procedure involving these data is completed. The work done by these authors is set out below in summary form.

Algorithm Online Sequential-Extreme Learning Machine: Given an activation function g or radial basis function kernel f, and hidden neuron or radial basis function kernel number Ñ for a specific application, the following two steps taken.

Step 1: boosting phase: Given a small initial training set ℵ = $\{(x_i, t_i)|x_i\epsilon\, R^m, t_i\epsilon\, R^m, i = 1, \dots .\, N\}$, the intention is to boost the learning algorithm by means of the following procedure:

(a) Assign random input weight w_1 and bias $b_1$or center μ_i and impact width $\sigma_i, i = 1, \dots, N$.
(b) calculate the initial hidden-layer output matrix $H_0 = [h_1, \dots, h_N]^T, where\ h_i = [(w_1 . x_i + b_1), \dots ., g(w_N . x_i + b_N)]^T, i = 1, \dots ., N$.
(c) Estimate the initial output weight $B^{(0)} = M_0 H_0^T T_0 where\ M_0 = (H_0^T H_0)^{-1}$ and $T_0 = [t_1, \dots, t_N]^T$.
(d) Set k = o.

Step 2: Sequential-learning phase: for each further incoming observation (x_i, t_i), where $x_i\epsilon R^n, t_i\epsilon R^m\ and\ i = N + 1. N + 2, N + 3, \dots .$, do the following:

(a) Calculate the hidden-layer output vector $h_{K+1} = [g(w_1 . x_i + b_1), \dots, g(w_N . x_i + b_N)]^T$.
(b) Calculate latest output weight $B^{(K+1)}$ based on a Recursive least square (RLS) algorithm: $M_{K+1} = M_k - \frac{M_k h_{k+1} h_{k+1}^T M_k}{1 + h_{k+1}^T M_k h_{k+1}}$
$B^{(k+1)} = B^{(k)} + M_{k+1} h_{k+1} (t_i^T - h_{k+1}^T B^{(k)})$
(c) Set k=k+1.

Extreme Learning Machine and Online Sequential-Extreme Learning Machine have been selected as the main classifiers for the power loss analysis to be undertaken in the present research. Both techniques were claimed to have well performance including to be extremely fast in its learning speed and to have better generalization performance when compared to conventional techniques. Unlike many more popular learning techniques, little human involvement is required in Extreme Learning Machine. Except for the numbers of the hidden neurons (insensitive to Extreme Learning Machine), no other characteristic need to be tuned manually by users because this techniques chooses the input weights randomly and analytically finds the output weights. Based on the above advantages, so far it is hard to point the disadvantages exists.

4.7.6 Support Vector Machine

Support vector machine has emerged as one of the most popular and useful techniques for data classification. The objective of Support Vector Machine [13] is to produce a model that predicts the target value of data instances in the testing set in which only attributes are given. For the sake of completeness, the fundamentals of the Support Vector Machine approach are reviewed briefly here.

The goal of Support Vector Machine is to estimate a function that is as close as possible to the target values $\mathrm{Y_i}$ for every $\mathrm{X_i}$ and, at the same time, is as flat as possible for good generalization. Given a set of data Points $(X_1, Y_1), (X_2, Y_2), \dots \dots . (X_1, Y_1)$, where $X_i \in X \subseteq R^n, Y_i \in R$ and l is the total number of training sample, Support Vector Machine approximates the function using the for lowing form $f(x) = w.\emptyset(x) + b$ Where, f (x) represents the high dimensional feature spaces which are non-linearly mapped from the input space x. The coefficients *w*and *b* is estimated by minimizing the following regularized risk function**:**

$$\frac{1}{2}||W||^2 + \frac{C}{l}\sum_{i=1}^{l}(\varepsilon_i + \hat{\varepsilon}_i)$$

There are two types of Support Vector Machine, Namely the linear Support Vector Machine and the nonlinear Support Vector Machine. The linear Support Vector Machine can occur in two cases, one involving separable data and the other involving non-separable data. The simplest form of Support Vector Machine classification is the maximal margin classifier. In a linear machine trained on separable data with training data $\{X_i, Y_i\}, i = 1, \dots \dots, l, Y_i \in \{-1,1\}, X_i, \in R^d$, suppose the hyper-plane that separates the positive and negative examples satisfies $W.X + b$=0. Where W is normal to the hyper-plane, b/||W|| is the perpendicular distance from the hyper-plane to the origin, and ||W|| is the Euclidean norm of W. This can be formulated as

$$X_i.W + b \geq +1 \; for \; y_i = +1 \quad \dots\dots\dots\dots\dots\dots\dots (4.4)$$

$$X_i.W + b \geq -1 \; for \; y_i = -1 \quad \dots\dots\dots\dots\dots\dots\dots (4.5)$$

And, can be combined into one set of inequalities.
The maximal margin is

$$\gamma = \frac{(X_1.W) + b}{||W||} - \frac{(X_2.W) + b}{||W||} = \frac{2}{||W||} \quad \dots\dots\dots\dots\dots\dots (4.6)$$

Therefore, the maximal margin classifier problem can be written in the following form: minimize $\frac{||W||^2}{2}$ $subject \; to \; Y_i\left(x_{i,} W + b\right) \geq 1, i = 1, \dots . . l$ The Lagrange multiplier method can be used to solve this optimization problem. In most real world problems, the training data are not linearly separable. There are two methods to modify the linear Support Vector Machine classification to suit the non-linear case. The first is to introduce some slack variables to tolerate some training errors so as to decrease the influence of noise in the training data. This classifier with slack variables is called a soft margin classifier. Consider the training data $\{(X_{1,} Y_1), \ldots\ldots, (X_1, Y_1)\}, x \in R^n, Y \in \{-1, +1\}$ with the assumption that they are linearly separable. That is, there exists a hyper-plane <W, X> + b=0 That satisfies the following constraints: for every $(X_i, Y_i), i = 1, \ldots., l$ $Y_i(< W, X_i >$+b)>0, where <W, X> is the dot product between W and X. The aim of the maximal margin classifier is to find the hyper-plane with the largest margin, that is, the maximal hyper-plane. This problem can be represented as: Minimize $\frac{||W||^2}{2}$ Subject to $Y_i, (< W, X, > +b) \geq 1, i = 1, \ldots., l$ In real-world problems, training data are not always linearly separable. In order to handle the non-linearly separable cases, some slack variable have been introduced into the Support Vector Machine so as to tolerate. Any training errors, with the influence of the noise in training data thereby decreased. This classifier with slack variables is referred to as a soft-margin classifier. A redial basis kernel function used in this experiment is of the form

$$K\ (X, Y) = exp\left(-\frac{\|X-Y\|^2}{2\sigma^2}\right). \quad \ldots\ldots\ldots\ldots\ldots (4.7)$$

4.8 Conclusion

In above chapter, developed unmetered power losses framework analysis that has been proposed for application in the prediction, detection and identification of unmetered power losses boned activities including power utility consumers. There are the five main modules to be represented like (i) Data Pre-processing (ii) Detection (iii) Feature selection (iv) Classification and (v) Prediction, all for detection, classification and prediction are moreover to be deliberated. The first step for the unmetered power losses is data pre-processing framework analysis proposed here. Several important factors that influence load data have been investigated, including time of consumption, weather, and social events. A data-selection method has been presented as the means of capturing the required electricity customer consumption data from the recorded meter readings. Next, the detection method to define customer behaviour according the designated categories has been developed. On the basis of pre-defined criteria, these customer behaviour categories have been established for use in the classification task.

This chapter presented the proposed unmetered power losses framework analytical and the key data-mining techniques to be used in unmetered power losses activity identification, detection, and prediction, including Extreme Learning Machine, Online Sequential-Extreme Learning Machine, and Support Vector Machine. Most of key flow processes in pre-labelling consumers' behaviour are also explain in this chapter, with a pre-processing activities to differentiated types of days on the basis of the raw data so as to normalise the data. It has been found that in this chapter consumers' load analysis and the correlation of such loads with weather, time and calendar-events factors.