

A SYSTEMATIC MACHINE LEARNING APPROACH TO SHORT TERM ELECTRICITY LOAD FORECASTING

DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
CONTROL AND INSTRUMENTATION

Submitted by:

Surbhi Singh
2K19/C&I/10

Under the supervision of

Dr. MADAN MOHAN TRIPATHI



DEPARTMENT OF ELECTRICAL ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of
Engineering) Bawana Road, Delhi-
110042

2021

DEPARTMENT OF ELECTRICAL ENGINEERING DELHI TECHNOLOGICAL UNIVERSITY

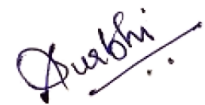
(Formerly Delhi College of
Engineering) Bawana Road, Delhi-
110042

CANDIDATE'S DECLARATION

I, Surbhi Singh, Roll No. 2K19/C&I/10, M.Tech. (Control & Instrumentation), hereby declare that the project Dissertation titled "A SYSTEMATIC MACHINE LEARNING APPROACH TO SHORT TERM ELECTRICITY LOAD FORECASTING" which is submitted by me to the Department of Electrical Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma, Associateship, Fellowship, or any other similar title or recognition.

Place: Delhi

Date: 10/09/2021



SURBHI SINGH

DEPARTMENT OF ELECTRICAL ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of
Engineering) Bawana Road, Delhi-
110042

CERTIFICATE

I, Surbhi Singh, Roll No. 2K19/C&I/10 student of M. Tech. Control and Instrumentation (C&I), hereby declare that the dissertation/project titled “**A SYSTEMATIC MACHINE LEARNING APPROACH TO SHORT TERM ELECTRICITY LOAD FORECASTING**” under the supervision of Dr. Madan Mohan Tripathi of Electrical Engineering Department, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology has not been submitted elsewhere for the award of any degree.

Place: Delhi
Date: 10/09/2021



Prof. Madan Mohan Tripathi
(SUPERVISOR)

Department of Electrical Engineering
Delhi Technological University

ACKNOWLEDGEMENT

I am highly grateful to the Department of Electrical Engineering, Delhi Technological University (DTU) for providing this opportunity to carry out this project work. The constant guidance and encouragement received from my supervisor Prof. Madan Mohan Tripathi of the Department of Electrical Engineering, DTU, has been of great help in carrying my present work and is highly acknowledged with reverential thanks. I would also like to thank my colleagues and classmates for their guidance and continuous support in the completion of this project work. Finally, I would like to express gratitude to all faculty members of the Electrical Engineering Department, DTU for their intellectual support throughout the course of this work.

SURBHI SINGH

2K19/C&I/10

M. Tech. (Control and Instrumentation)

Delhi Technological University

ABSTRACT

In the energy sector, for an efficient electricity load management which includes viable utilization and allocation of energy assets, Electricity Load Forecasting plays a critical role. Precise long-term and short-term electricity demand forecast is significant as it enables complete utilization of produced electric power, preventing over-production and sometimes wastage of energy and resources. short term load forecasting is necessary, as it is used to maintain and regulate the optimal performance in the daily operation of electrical power systems. With enhancement in technology and automation, and the rise of artificial intelligence it becomes an eminent need to bring about revolutionizing changes for forecasting the future energy needs. In this regard researchers are trying to explore, study, innovate and improving the existing machine learning and deep learning methodologies for the purpose of accurate and efficient electricity load forecasting. Hence through this study, Ensemble based methods and neural network-based methods are explored and the results have been provided for comparative evaluation.

This thesis presents a comparative proof of ensemble learning based algorithm Extreme Gradient Boosting Technique (XGBoost) with Deep Recurrent Neural Network (RNN) and Stacked Long Short-Term Memory Network (LSTM) for short term electricity demand forecast on the Dominion Energy Data taken from PJM energy market. The aim of this thesis is to investigate and prove that stacked LSTM performs better as compared to an ensemble machine learning model XGBoost and deep RNN algorithms on PJM energy data, by using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R2 score as evaluation metrics for performance validation. Here in this study, RNN, stacked LSTM model and XGBoost model is compared with a Hypertuned stacked LSTM model. The Hypertuned model which had increased number of hidden nodes from the initial LSTM network tend to give an improved performance, proof of which is shown through this study. This work sheds light on the internal architecture of the models and the different values of hyper-parameters used while training the models to justify the observed day-ahead predictions.

CONTENTS

CANDIDATE’S DECLARATION	ii
CERTIFICATE	iii
ACKNOWLEDGEMENT	iv
ABSTRACT	v
CONTENTS	vi
LIST OF FIGURES	viii
LIST OF TABLES	ix
CHAPTER 1	10
INTRODUCTION	10
1.1 ELECTRICITY MARKET REPRESENTATION:	10
1.1.1 REGULATED ELECTRICITY MARKET	10
1.1.2 DEREGULATED ELECTRICITY MARKET.....	11
1.2 ELECTRCITY DEMAND.....	12
1.3 ELECTRCITY DEMAND FORECASTING	12
1.3.1 TYPES OF LOAD FORECASTING.....	13
1.4 IMPORTANCE OF DEMAND FORECASTING.....	14
1.5 CHALLENGES FACED DURING DEMAND FORECAST	15
1.6 OUTLINE OF THE THESIS	15
LITERATURE SURVEY	17
2.1 OVERVIEW	17
2.2 LITERATURE REVIEW	17
2.3 CONCLUSION.....	18
CHAPTER 3	19
DATASET AND DATA-PREPROCESSING	19
3.1 DATA COLLECTION	19
3.2 FEATURE ENGINEERING	19
3.3 DATA NORMALIZATION.....	20
3.4 DATA DIVISION.....	21
CHAPTER 4	22

METHODOLOGY AND CONFIGURATION FOR SHORT TERM ELECTRICITY LOAD FORECASTING	22
4.1 OVERVIEW	22
4.2 RECURRENT NEURAL NETWORK (RNN)	23
4.3 EXTREME GRADIENT BOOSTING (XGBOOST)	24
4.4 LONG SHORT-TERM MEMORY (LSTM) NETWORK	27
CHAPTER 5	31
RESULTS AND DISCUSSION	31
5.1 EXPERIMENTAL SETUP	31
5.2 MODEL PARAMETERS	31
5.3 SIMULATION RESULTS	31
5.3.1 RNN	33
5.3.2 LSTM MODEL 1	34
5.3.3 LSTM MODEL 2	35
5.3.4 XGBOOST	36
5.4 PERFORMANCE METRICS	37
5.5 RELATIVE ERROR PLOTS	38
CHAPTER 6	41
CONCLUSION & FUTURE SCOPE	41
REFERENCES	42
APPENDIX-I	45
APPENDIX-II	52

LIST OF FIGURES

FIG. 1.1 REGULATED ENERGY MARKET.....	11
FIG. 1.2 DEREGULATED ENERGY MARKET	11
FIG. 1.3 VARIATION OF ENERGY WITH TIME [1]	12
FIG. 1.4 TYPES OF ELECTRICITY LOAD FORECAST	13
FIG. 3.1 ELECTRICITY DEMAND ON THE ORIGINAL SCALE AGAINST DATE	20
FIG. 3.2 ELECTRICITY DEMAND DATA AFTER NORMALIZATION AGAINST DATE.....	20
FIG. 4.1 TYPES OF MACHINE LEARNING ALGORITHM.....	22
FIG. 4.2 ARCHITECTURE OF FEED FORWARD NETWORK.....	23
FIG. 4.3 ARCHITECTURE OF RNN	24
FIG. 4.4 WORKING OF A BAGGING ALGORITHM	25
FIG. 4.5 WORKING OF A BOOSTING ALGORITHM.....	26
FIG. 4.6 WORKING OF XGBOOST [24]	26
FIG. 4.7 WORKING OF A LSTM UNIT [28]	29
FIG. 5.1 TWO WEEKS PREDICTION PLOT FOR SIMPLE RNN MODEL.....	33
FIG. 5.2 SINGLE DAY HOURLY FORECAST BY RNN MODEL	34
FIG. 5.3 TWO WEEKS PREDICTION PLOT FOR LSTM MODEL 1	34
FIG. 5.4 SINGLE DAY HOURLY FORECAST BY LSTM MODEL 1	35
FIG. 5.5 TWO WEEKS PREDICTION PLOT FOR LSTM MODEL 2	35
FIG. 5.6 SINGLE DAY HOURLY FORECAST BY LSTM MODEL 2	36
FIG. 5.7 TWO WEEKS PREDICTION PLOT FOR XGBOOST MODEL.....	36
FIG. 5.8 SINGLE DAY HOURLY FORECAST BY XGBOOST	37
FIG. 5.9 PLOT OF RELATIVE ERROR FOR RNN MODEL PREDICTIONS	38
FIG. 5.10 PLOT OF RELATIVE ERROR FOR LSTM MODEL 1 PREDICTIONS	39
FIG. 5.11 PLOT OF RELATIVE ERROR FOR LSTM MODEL 2 PREDICTIONS	39
FIG. 5.12 PLOT OF RELATIVE ERROR FOR XGBOOST MODEL PREDICTIONS.....	40

LIST OF TABLES

TABLE 5.1 TRAINING PATTERN FOR RNN MODEL	32
TABLE 5.2 TRAINING PATTERN FOR LSTM MODEL 1	32
TABLE 5.3 TRAINING PATTERN FOR LSTM MODEL 2	33

CHAPTER 1

INTRODUCTION

1.1 ELECTRICITY MARKET REPRESENTATION:

Electricity is effectively distributed to every sector in the world and delivered to consumers from power generators, through transmission cables. The system for mitigation and management happens via an electricity or energy market. Electricity being a commodity which happens to be tradable and allows variables to differentiate itself from others for example, crude oil or natural gas. It can be also said that one megawatt hour of electricity produced from either coal or natural gas contains exactly the same amount of energy. It is bought and sold like many other types of commodities. It must be produced and used simultaneously.

It also no news that there are a lot of participants in the energy market, which lead to competitive pricing. The energy or electricity market as researched and observed, cost-effectively aligns the ability to provide electricity, or supply, with our need for electricity, or demand which varying through the entire day.

The supply must meet demand exactly. And services such as power grid ancillary and demand response programs enable and manage that increase in demand and effectively handling demand surges in real time. Meaning generation resources are not enabled during low consumer demand, but only during peak consumer demand periods. One of the responsibilities of Electricity System personnel, is to forecast the amount of electricity based on the trend followed by the consumers demand throughout a time duration.

1.1.1 REGULATED ELECTRICITY MARKET

In this classification of electricity market, powerhouses that are vertically coordinated occupy the whole value chain regulated by a public controller. The utility ensures that the generated electrical power is transferred to the grid, and then to the consumers. Consumers in this kind of markets do not have the freedom to choose who is the generation powerhouse and hence are bounded to the powerhouse in that area. From the generation to the meter, the utility has complete authority. In a regulated market setup, there I stop to down approach while starting from generators, then to transmission to distribution and finally to the consumer side, as can be seen from the fig. 1.1.

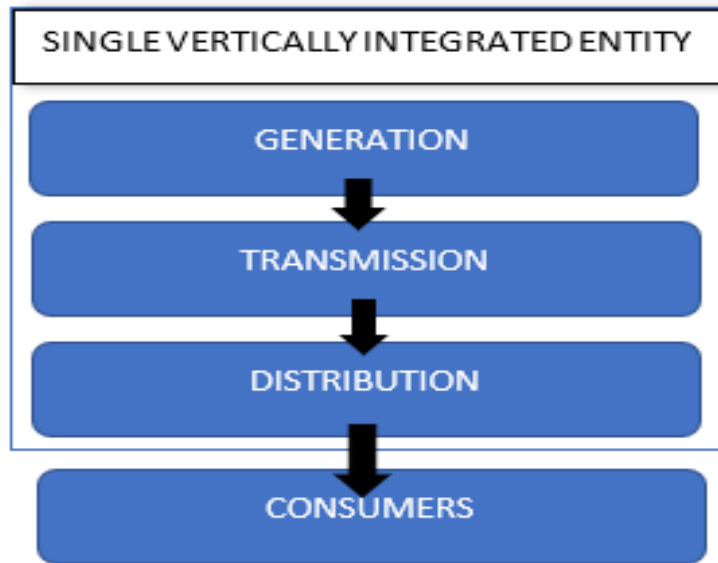


Fig. 1.1 Regulated Energy market

1.1.2 DEREGULATED ELECTRICITY MARKET

A “deregulated electricity market” is basically a liberated power market, which considers for the entrance of contenders to purchase and sell electric power by allowing market participants to put resources into power plants and transmission lines. Generation proprietors then offer this discount power to retail providers. Retail power providers set costs for customers, which are frequently alluded to as the "supply" segment of the power bill. Energy deregulation allows consumers and business owners to choose their energy supplier and it couldn't be easier there's no service disruption reliability and service is unchanged.

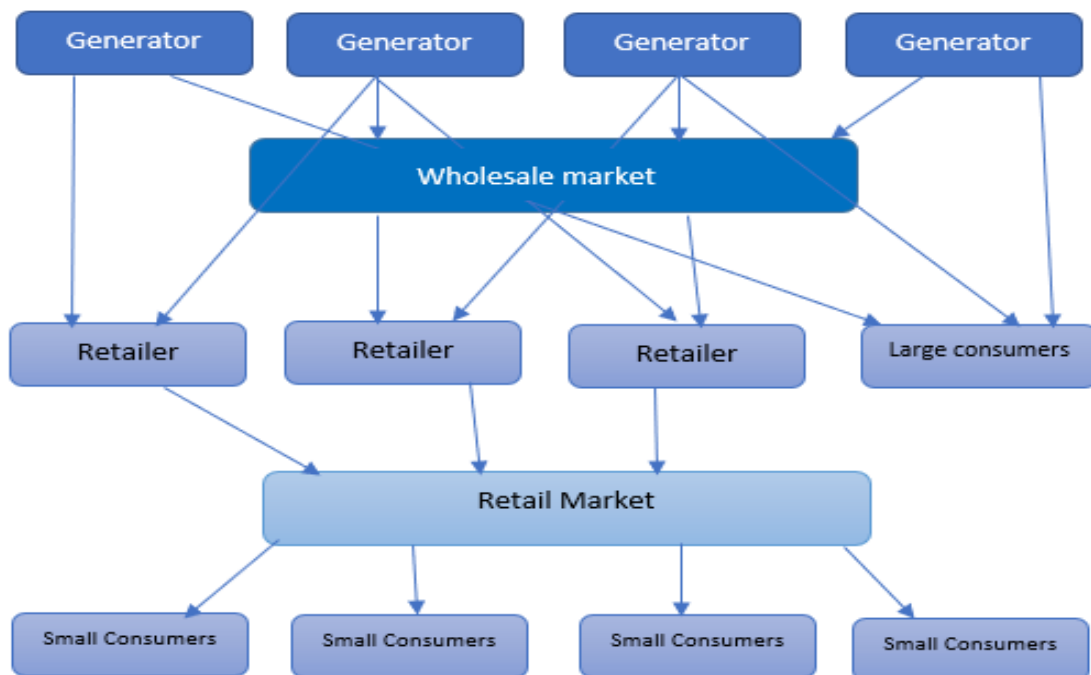


Fig. 1.2 Deregulated energy market

1.2 ELECTRCITY DEMAND

Electrical Energy is an eminent energy resource which is being utilized everywhere in the world on a standard premise as it helps power the gadgets, and run industries and livelihood. Electricity being one of the most important and vital inventions till date since it fills in as the standard for all developments to come, Electrical Energy has become a significant asset in this advanced age.

What is demand? What's the difference between kilowatts and kilowatt-hours? Can be just as important as how much electricity one uses. If we are on a time of use rate, our bill is based on how much energy we use, when we use it and, for some customers, their maximum demand or load, or how much is used at any given time. The maximum load at any point in time is called Electric Demand and it is measured in kilowatts. The amount of electricity used over a period of time is measured in kilowatt hours. It is also important to note that, the more electricity used at one time, the higher the demand and the harder the grid has to work to meet customers' needs. The load demand is seen to vary somewhat like the fig 1.3.

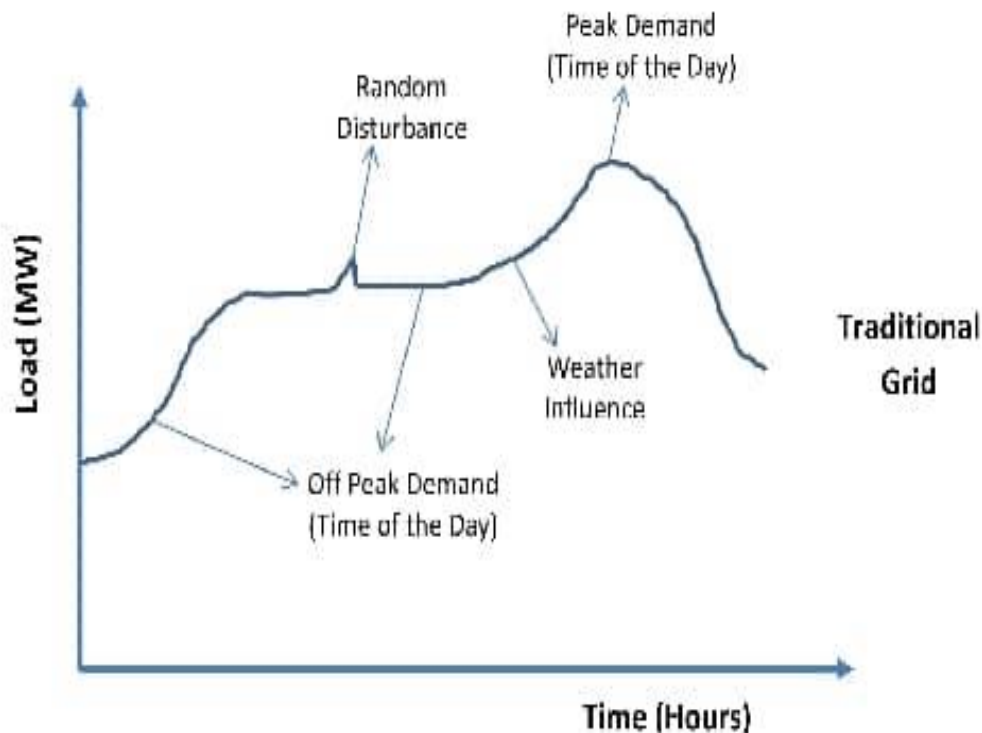


Fig. 1.3 Variation of energy with time [1]

1.3 ELECTRCITY DEMAND FORECASTING

In the electrical power sector, the electricity load prediction or forecasting can be understood as the strategy with the help of which Energy Companies foresee the electricity demand needed by the customers be it at a residential level or at a commercial level and then accordingly, they fulfill the necessary electricity demand.

For optimal power system operation, electrical generation must follow electrical load demand. The power companies need certain mechanisms to predict the future consumer demand so they can use the electrical infrastructure effectively and efficiently with no

financial losses. In this reformist era of innovation, where knowledge of historical data is key in improving the future prospects of outcomes in any domain of industry, Energy Sector is no exemption.

The main purpose of forecasting is to meet future requirements, reduce unexpected cost and provide a potential input to decision making.

1.3.1 TYPES OF LOAD FORECASTING

A specific forecasting procedure may be classified as below, depending on the time period of interest,

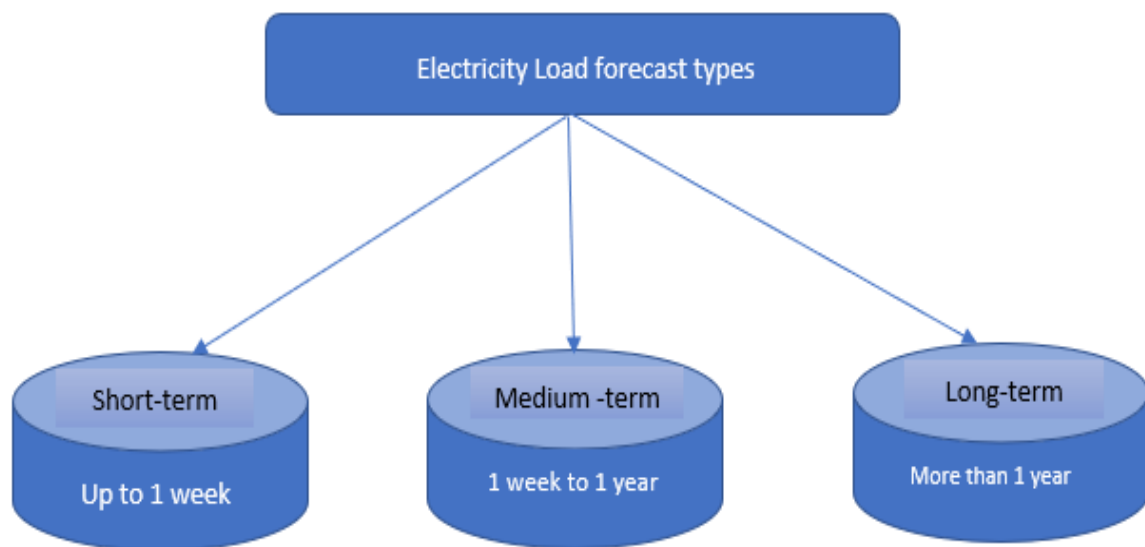


Fig. 1.4 Types of Electricity Load forecast

SHORT- TERM LOAD FORECASTING

As the name suggests, in short term load forecasting, the purpose is to predict the electricity demand for a shorter duration of time such as for the upcoming one hour up to the next two weeks or from one hour to one week. Short term Load forecasting is calculated to as they are critical for daily functioning of a power system starting from economic load dispatch and electricity load management. This kind of demand forecast value in other words also called an hour-based electricity load forecast. For day-to-day operation, covering one day or a week, Short-Term forecasting is needed in order to commit enough generating capacity formatting the forecasting demand and for maintaining the required spinning reserve.

MEDIUM - TERM LOAD FORECASTING

Medium-term electricity forecasts give predictions from a few weeks to a year. They are helpful in strategizing fuel procurement, for load scheduling unit and planning energy trading and assessment of the revenue for the utilities. A medium-term forecast is also called as the monthly load forecast. The methodologies utilized for this type of classification of forecast are same as used for the short-term forecasting technique except that there is less need for accuracy [2].

LONG - TERM LOAD FORECASTING

The prediction of Long-term electricity demand is known to be an important estimation in the electricity demand strategizing, it is very helpful in regulating and managing the tariff and also in energy trading. This kind of forecasted values are used to determine the descriptive power sector electricity generation blueprints and plans for electricity transmission plans. The long term forecasted value is usually referred to as an annual peak load. Long Term electricity Forecasting is done for maintenance plans of the generating plants, to lay the future increase of the generating capacity, and to make agreements for energy exchange with subordinate or neighboring generating units. Generally, the techniques used for long-term forecasting other than regression algorithms are expert system methods such as genetic algorithms.[3]

1.4 IMPORTANCE OF DEMAND FORECASTING

Load forecasting be it long-term or short-term, is one of the important and pioneering areas of research, as predicting the probable electricity consumption not only helps to create a balance between the electricity demand and supply but also helps power plants in better load scheduling thus preventing excessive production of electricity. Predicted results are also vital in the contingency planning and strategizing and economic load dispatch. The electricity demand is changing with time; hence the supply needs to be adaptable to this dynamic behavior. Some of the advantages of electricity load forecasting are listed below:

- Forecasting, enables the utility companies to plan well and allocate power sources in advance since they have an understanding of the future consumption or load demand.
- Limit financial risks for the power companies. Having knowledge of the future electricity demand assists the companies to dwell into monetarily practical choices when it is about the upcoming electricity generation and transmission projects.
- Assists with deciding the necessary assets which such as fuel for utilization in operation of the electricity generating plants as well as other resources that in future are bound to guarantee continuous and moreover a very efficient generation and electrical power distribution to the end users.
- The electricity demand/load prediction is also helpful in planning the location, size and kind of the power plant. Through recognizing the specific regions with high or developing electricity demand, the power companies will plan on generating the power close to the load.
- Required for the Planning and strategizing of the regulation of the utility frameworks. With this knowledge of future demand, the utility is in the condition to realize when to start with the correction and maintenance and guarantee so that it minimally affects the consumers. For instance, they may schedule maintenance in regions demand is very low.
- Forecasting helps proper utilization of electricity produced by the powerhouses hence, avoiding over the limit or less production of electricity.

1.5 CHALLENGES FACED DURING DEMAND FORECAST

- Forecasting is also dependent on natural conditions like weather. As, the weather can sometimes be unusual the estimation of electricity demand may hence become different when the actual weather contrasts from anticipated one.
- Varied regions are bound to encounter variable weather conditions that can at a certain extent influence the energy load from the consumer's side. This will at the next point contrarily affect revenues, particularly assuming that the powerhouses satisfy generate a higher amount to make sure they fulfill the expected high demand.
- A vast majority of the extremely successful and accomplished utility forecasting members utilize strategies which are manual that depends upon an exhaustive comprehension and understanding of a broad spectrum of influencing factors dependent on forthcoming occasions or in other scenarios as specific data which is depending on the manual anticipating isn't reasonable because of the expanding size of data and intricacies involved in forecasting.
- The utilization conduct fluctuates between buyers utilizing various kinds of meters and particularly varying between smart and traditional meters and also different tariffs. The utility should understand this and foster separate models for each of the metering systems and then give out the averaged results, else erroneous predictions may happen.
- At times, it is difficult to get exact dataset for the popular utilization nature because of the fluctuations in noteworthy points, for example, electricity rates and the power demand based on any changes in rates.
- The predicted demand may sometimes vary for two seasons because of the known and anticipated complex behavior of consumer's demand, as consumption is different if the effect of two season is different.
- It is on some instances difficult to consider the variables that influence consumer's demand into the forecasting mechanisms. Also, forecasted demand value can sometimes be inaccurate when demand is dependent onto variables for example, change in temperature, increase or decrease in air humidity, solar exposure, etc., which affect energy usage as the model get complex.
- The utilities may incur losses if a threshold to limit of error in electricity demand forecasting is not pre decided.

1.6 OUTLINE OF THE THESIS

This thesis has been divided into following sections:

1. Chapter 1 gives the brief introduction to electricity load forecasting, an insight into energy market, types of load forecasting techniques, its importance along with the challenges.
2. Chapter 2 provides a gaze into the literature survey on the existing research available on electricity load forecasting.
3. Chapter 3 informs about the dataset and its processing which highlights data normalization and training and testing splits of data set.
4. Chapter 4 explains the architecture and working of LSTM, RNN and XGBoost used for predictive analysis.

5. Chapter 5 presents the plots showing the predicted values with the target values, extracted after predictive analysis. This chapter also contains the obtained values of evaluation metrics for RNN, LSTM model 1, LSTM model 2 and XGBoost in a comparative manner.
6. Chapter 6 concludes the thesis after the investigative and evaluative study of the Electricity load forecasting with Machine learning and deep learning algorithms.

CHAPTER 2

LITERATURE SURVEY

2.1 OVERVIEW

This chapter gives a short overview of the previous research done for Electricity Load forecasting using various conventional and non-conventional techniques. The literature survey provides a peek into various methodologies considered for prediction problems based on regression.

2.2 LITERATURE REVIEW

In today's period of development, the most pioneering aspect in any domain of technology is learning from the patterns of the past and giving out result in the present so as to create a sustainable and an efficient innovation space. With the energy sector being no exemption, short-term forecasting of electricity load, based on historical load data, becomes a spearheading space of research. Gauging the electricity demand is a primary concern as it assists in creating a balance between the electricity consumption and its supply.

Effective load scheduling and designing load dispatch strategies is possible because of short term and long-term electricity demand forecast. Accurate predictions prevent financial crisis for the stakeholders [4]. Also, by utilizing these predicted load data points, allocation or power sources can be done, when there is a peak demand and in times of low demand without over generation or under generation, by power grids but optimum production. Hence in order to avoid inaccuracies in the forecasted load values and encourage precision, researchers have been studying and analyzing the results of various methodologies.

These techniques can be classified in the broad categories of conventional and AI-based methods. Traditional or conventional techniques include mathematical models, including multiple linear regression [5], stochastic time series [6], exponential smoothing [7], knowledge-based Expert methods [8] and statistical approaches such as ARIMA. The AI-based category for prediction methods takes fuzzy logistic methods [9], neural networks (NNs), neuro-fuzzy systems and Support Vector Machines (SVMs) [10] and their hybrid models under its umbrella. Majorly used among these methods are NNs [11]. AI methods evidently tend to perform better as they take care of the non-linear behavior in the electricity demand recorded over time [12].

Some techniques like Random Forest, Bagging and M5P which are regression tree-based machine learning algorithms are utilized to efficiently perform electricity forecasting. The advantage of these techniques was verified through broad tests on the load data from the Australian power market and M5P gave desirable results [13]. The Support vector regression technique has also proven to perform better while predicting when features like weather, calendar, and holiday data, are included in the feature matrix [14].

Short-term load forecasting, has time also acting as a variable. Hence statistical techniques like ARIMA and neural networks, were used for time series forecasting on the Kuwait dataset using neural networks and ARIMA was done and weather data such as temperature

measures and humidity measure, oil price data, population, residence count, average salary, number of passengers, rate at which currency was earned, and economic factors were utilized as features, and neural networks proved to perform better than ARIMA [15]. It also observed that the artificial neural network model is 10% more accurate than conventional methodologies [16]. In [17] ARIMA was combined with SVM and Neural Networks for predictive analysis.

Data also plays an important role in churning out accurate models, as some investigations also suggest that a primary requirement for any prediction methodology is to have an adequate amount of historical data for training. This will improve the testing accuracy as well as the precision resulting in reduced error between target and forecasted value and the model is able to avoid underfitting. It is also beneficial if multivariate models are built where features incorporated can be consumer behavior, commodity price of other sources such petroleum, oil, etc., appliances measurements and weather features, in order to enhance accuracy [18]. Another class of prediction model which is expected to perform better is Extreme Gradient Boosting technique or XGBoost. Simulation results in certain studies show that the XGBoost model was able to give out prediction of the electricity load effectively due to its features of regularization and handling of sparse data [19].

More advanced methods which come under the category of deep learning models were put to use for electricity demand forecasting. One of them was Recurrent Neural Network [20]. To overcome non-stationary and long-term dependencies challenges, long short-term memory (LSTM) [21], was developed and used in many prediction problems.

2.3 CONCLUSION

Based on the literature survey, this work is demonstrating three models XGBoost, RNN and stacked LSTM and Hypertuned stacked LSTM for future load prediction, and is aimed at comparing their statistical metrics calculated for model performance evaluation and deciphering the reason for their behavior of prediction and how neurons-based network outperforms the infamous ensemble learning based boosting technique.

CHAPTER 3

DATASET AND DATA-PREPROCESSING

3.1 DATA COLLECTION

For any predictive modelling problems, the machine learning models use ample rows of Data to get trained and make predictions. Data gives information about the Dominion Energy Inc. distribution, which supplies power in Virginia, North Carolina, and South Carolina. It is extracted from PJM energy market and is obtained from Kaggle [22]. The dataset contains per day hour-based electricity demand in unit of Mega Watts starting from May 2005 to August 2018. The data has total 116189 number of rows, where for a single day, there are 24 records of electricity demand.

3.2 FEATURE ENGINEERING

Since the data that is collected is usually unclean, it becomes necessary to perform data wrangling before predictive modelling. First, the data is checked for any missing value. Then the data type of each variable use is checked. The variable sin a data is usually classified into the following categories:

- **Independent Variable / Feature matrix:** It is a variable that is not affected or changes when there is a change in other variables that is to be measured. It is also the variable that is controllable variable which tests the effects or changes on the dependent variable. For example, in this study, variables like Weekday, day, hour, Lagged electricity Load, etc. has no effect on one other although they affect the dependent variables that are in original electricity demand data points that needed to be forecasted.
- **Target variable:** This is a variable that depends on the feature matrix within the data set. This is the variable which is the target that is to be predicted or tested. Such as, the actual demand data points from the data that needs to be forecasted is the dependent factor that is influenced by the other variables of the feature matrix.
- **Categorical Variable:** Categorical variables are those type of variable which take values that can be put in a category. They can be ordinal and nominal, where ordinal follows an order such as low, medium, high and nominal does not follow any order like names of states.

In this work the categorical variables which are used are Hour of the Day, weekday, etc. are all categorical variables that are not numerically significant.

- **Quantitative Variable:** These are types of variables which are numerical. They are quantities that can be measured. These values have mathematical significance. Example of the quantitative variable in this study is the Lagged Load and the real electricity consumer's load all go to the above-mentioned category.

Hence in this research work, the data type of the date time column and electricity demand column was checked and changed to the format compatible for a machine learning model. Dataset was checked for the missing values. New columns were created using the date-time column, which are the day of the year, day in the month, day in the week, hour,

month, year, which were used for creating a feature matrix in case of XGBOOST algorithm.

3.3 DATA NORMALIZATION

Dataset normalization is the method to structure a dataset in a way, so that data points redundancy is reduced and integrity of each record in the dataset is improved. Normalization is principally done to bring every data element on a comparative scale so that every component is similarly critical. Also, as suggested in [23], the normalization techniques influence execution of a model at a principal level in prediction problems which use neural networks as models, bringing about better precision. Of the many normalization techniques MIN_MAX normalization is used in this work.

As evident from the figs. 1 and 2, MIN_MAX normalization is utilized to transform the power consumption in the original data from the scale of 5000 MW to 22500 MW to a normalized scale in the range of 0 and 1.

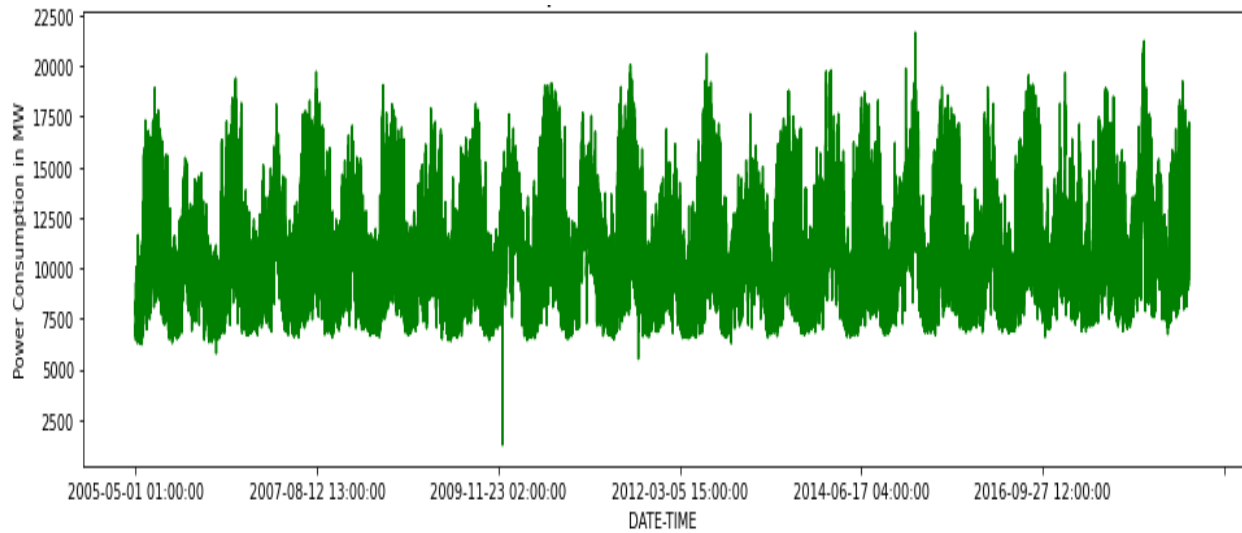


Fig.3.1 Electricity Demand on the original scale against Date

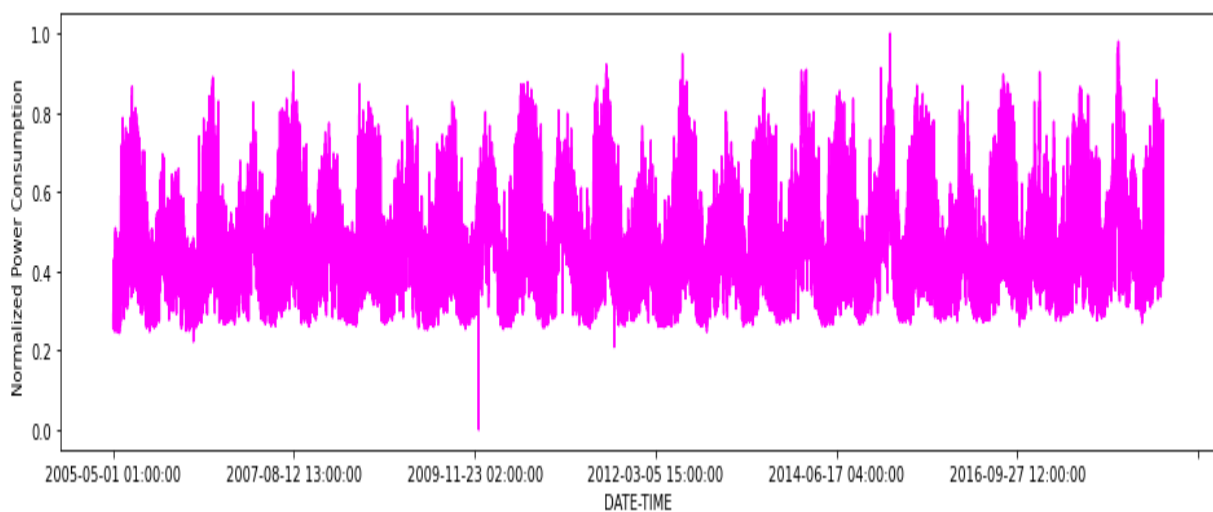


Fig. 3.2 Electricity Demand data after Normalization against Date

This normalization technique changes data points from a point of [MinT, MaxT] to a point of [MinN, MaxN] dependent on the greatest estimation (MaxT) also, on the least estimation (MinT) of the original dataset. Subsequently for each feature variable, the least value of that feature in consideration gets changed into a 0, the greatest data point gets changed into a 1, and each and every other value gets changed into a decimal somewhere in the range of 0 and 1. Mathematically, it is represented by eq. 3.1 below.

$$N_i = \text{Min}_N + \frac{T_i - \text{Min}_T}{\text{Max}_T - \text{Min}_T} \times (\text{Max}_N - \text{Min}_N) \quad (3.1)$$

Here, N_i is the normalized value.

3.4 DATA DIVISION

Before approaching the stage of predictive analysis, there is a need to divide the dataset into two different splits. One is called the training split and the other is known as the testing split. A machine learning model is never tested on the trained data. Through the training set the model learns the pattern and difference in each data record and figures out how is the feature matrix influencing the target and tries to make use of this learned pattern to get assessed on an unseen data (testing split). Hence the percentage of training split is more than the testing split. Following which, in this paper 80% of data i.e., 92951 rows of data compose the training set and rest 20% which are 23218 records used for testing the models.

CHAPTER 4

METHODOLOGY AND CONFIGURATION FOR SHORT TERM ELECTRICITY LOAD FORECASTING

4.1 OVERVIEW

Every machine learning model or algorithm can be categorized into below categories:

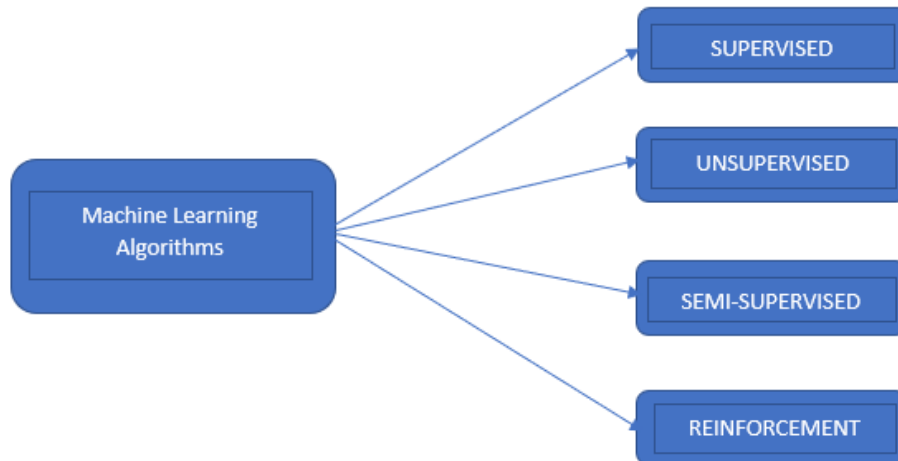


Fig. 4.1 Types of Machine Learning Algorithm

SUPERVISED ALGORITHMS: This type of algorithm takes a label or target to predict. Here algorithm attempts to create relationships and dependencies between the target matrix and the input feature matrix, so that it can predict the output for new unseen data based on those relationships and patterns that it learned from the initial data set. It can be said that supervised machine learning algorithms require a “teacher” in the form of target to signify what needs to be predicted.

UNSUPERVISED ALGORITHMS: Are the category of ML algorithms which are generally used in pattern detection. Here there are no labels or target matrix provided to the model, based on which it could try to create relationships between feature and target values. These algorithms use techniques to decide for rules, detect patterns, and group the data points on their own to extract meaningful insights and describe the data better to the users. Hence they do not require a “teacher.

SEMI SUPERVISED ALGORITHMS: In the previous two kinds of algorithms, one takes no target for the features in a dataset and the other contains target matrix or label along with the feature matrix. Semi-supervised Algorithms is somewhere in between the above two. In numerous functional circumstances, the expense to assign label is high, as it requires specialists that are skilled. So, semi-supervised algorithms are the best contender for the model structure when there is absence of labels in most of the dataset and present in few. These methods work best even when there are observations with absent labels as this data provides important information about the group.

REINFORCEMENT LEARNING: In Reinforcement learning a machine learning model Is trained to make a sequence of decisions. The model learns to get to a target in an unsure, conceivably complex environment. Here the model goes through sequence of trial and error to solve a problem. Following which there is reward or penalty associated with every action of model. So, for every desirable output the model is rewarded and it is penalized for every wrong output. The model learns on its own how to maximize the rewards.

This research work explores 3 kinds of algorithms, Recurrent Neural network, XGBoost and LSTM. Of the three XGBOOST is a supervised machine learning algorithm whereas RNN and LSTM are used as Semi supervised algorithms since they used lagged values of load values as target.

4.2 RECURRENT NEURAL NETWORK (RNN)

RNN are a category of neural nets which in most situations are beneficial and more accurate in modelling the sequential information. They are derived from feed-forward networks. To understand the recurrent neural network, it is important to get an idea about the Feed forward network.

In a feed-forward neural network (FFNN), the data from the first neuron is transferred in forward fashion beginning from the input layer, passing from and within the hidden layers, and finally to the output layer. As mentioned in previous step, that the data goes straight in a forward fashion, which means it never contacts a node more than once, as evident from the fig. 4.2. Due to this unidirectional behavior, they have no memory of the previous input data they get and only consider the current input; hence they are terrible at anticipating the future,

Feed forward neural networks are mostly used f in situations where the information to be learned is not successive.

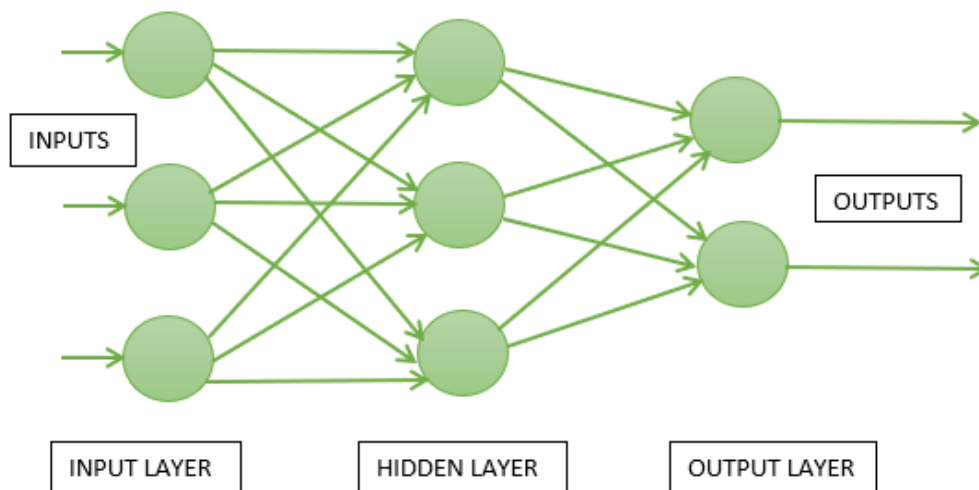


Fig. 4.2 Architecture of feed forward network

A FFNN, as suggested in previous and primary other deep learning algorithms, provides a weight vector to each of the neurons in the input layer and hence gives out the outcome. Whereas the RNNs provide weights to the neurons in the current as well as to the neurons in the prior input layer. Hence, a RNN has the ability to memorize the characters because

of its memory element. It gives out outcomes, then takes this outcome and is following a structure fed back into the architecture. So, from the architectural perspective, RNNs belongs to the class of artificial neural networks where, data enters one from one layer of neurons to another with feedback where the new output in the model is influenced by its preceding states. Since, RNNs are different from feed forward neural networks, as the they have a memory element which allows it to store data from its past calculations unlike the feed forward networks which are stateless, as can be seen from the figure below.

Some of the prominent applications of RNNs are language modelling, speech recognition and reinforcement learning due to their dynamic temporal behavior. The feedback element adds to the advantage of the network as it improves the learning capacity to give more accurate outcomes.

In a recurrent neural network, the decision made by the recurrent network at some time instance signified by t , will be influenced by the decision made at the time instance denoted by $t-1$. Structurally, RNN unit has 2 input values, the input x_t from present time instant and the preceding hidden state which is represented by ' h_{t-1} ', (output from the past state). The mathematical intuition behind this can be given by eq. 4.1

$$h_t = f(W * x_t + U * h_{t-1}) \tag{4.1}$$

Here $f(\cdot)$ = activation function

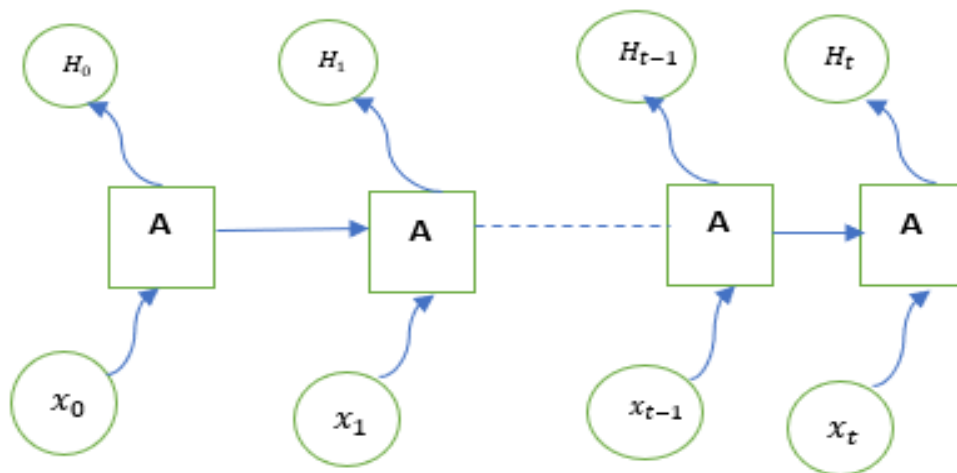


Fig. 4.3 Architecture of RNN

4.3 EXTREME GRADIENT BOOSTING (XGBOOST)

At times, it may not be adequate to depend upon the results of only one ML model. Ensemble learning provides a perfect solution to consolidate multiple learners together and give the resultant prediction which is the aggregated output from these combined models.

The operation of Ensemble models is categorized in the following two:

BAGGING

Bagging, follows bootstrap accumulating. It is the process wherein numerous similar learning algorithm are trained with samples which are bootstrapped versions of the original

data. Here bootstrapping signifies a statistical procedure for assessing metrics about a population by averaging these estimates from varied small data samples.

Decision trees come under this category. Decision trees are supposed to have high variance, hence bagging aggregation aids in decreasing the variance in the base learners. In bagging technique, many decision trees are generated in parallel, and these decision trees form the base learners. The bootstrapped Data samples are fed to the base learners for training. For the final prediction, output from all the base learners is the averaged, as can be shown from the fig. 4.4

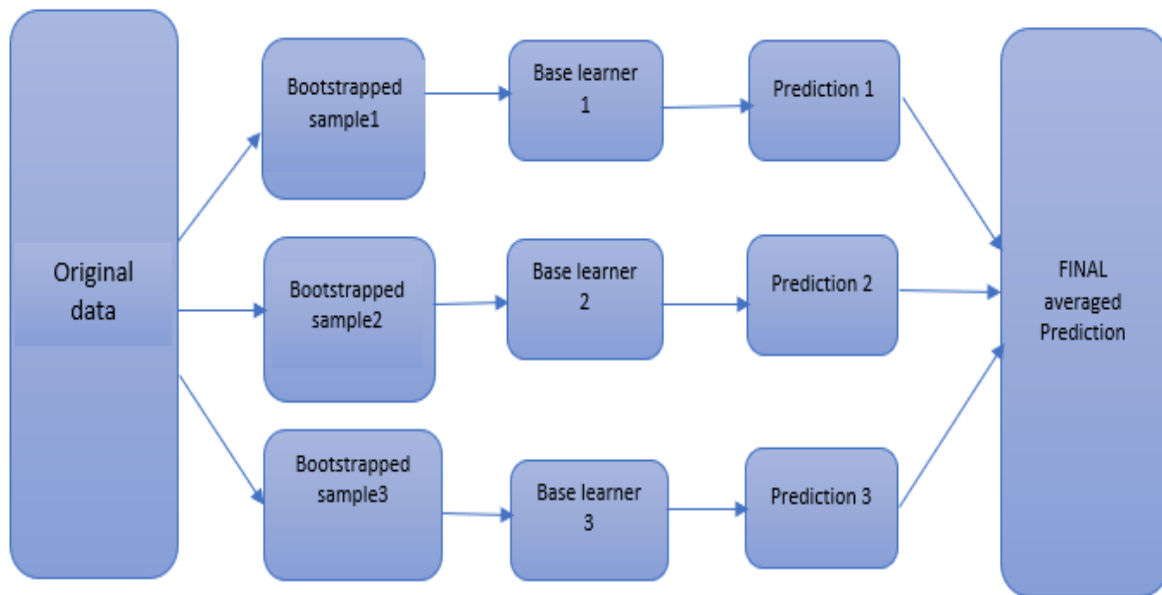


Fig. 4.4 Working of a bagging algorithm

BOOSTING

Boosting is a conceptual and architectural variant of bagging. Here at the backend small decision makers are built separately and in a manner of sequence, counterfeiting the previous one. So when a data value is falsely classified in a wrong group or inn general, then the next decision maker or model corrects it. This happens sequentially thus degrading the erroneous classification or prediction and boosting the correct the grouping.

Each decision model is trying to learn from the mistakes of the past models and finally updating cumulative errors as seen in fig.4.5. So the incoming model in the sequence will an updated one.

The decision models under the hood in boosting technique learners in which the bias is high, and the accuracy for regression or classification is just a small amount in the right direction. Every decision model below is important as they provide critical and notable description or behavior about the prediction, and which in return forms an accurate decision base model by superimposing the past models.

Showing alternative behavior from to bagging algorithms, where trees are created at as many branches as possible, boosting algorithms create decision models with comparatively less branches, so they can be dived into easily.

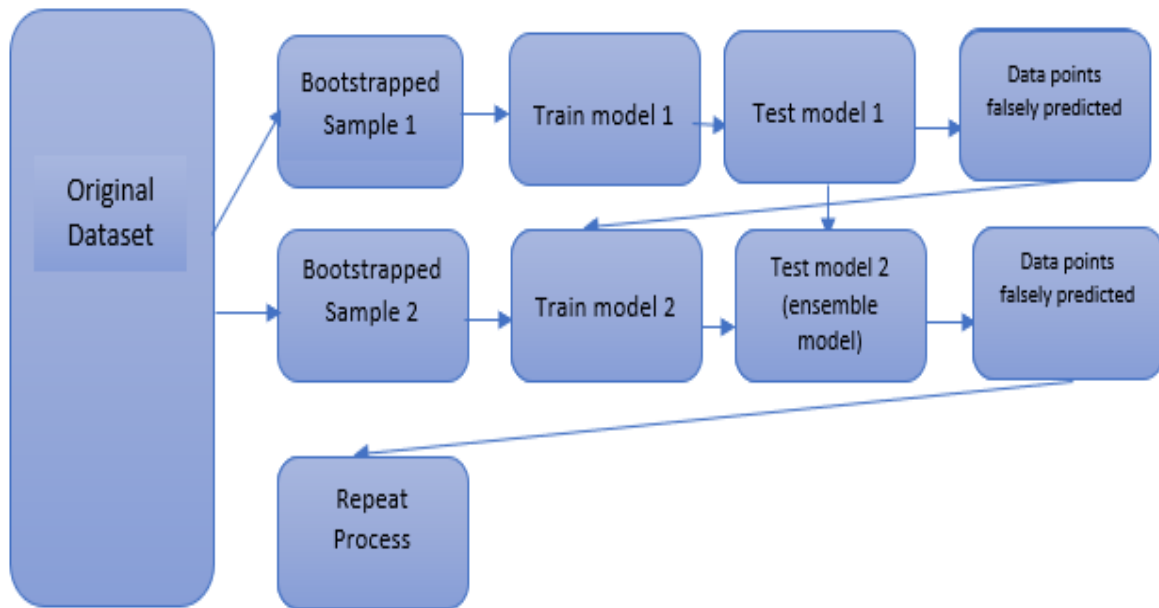


Fig. 4.5 Working of a boosting algorithm

XGBoost belongs to the category of the boosting algorithms of the ensemble machine learning models. The ensemble learning basically combines the outputs of the base learners with high bias that run under the hood, the tree-based machine learning algorithms. In boosting, the base model is constructed consecutively to such an extent that each resulting model intends to decrease the blunders of the past model. Each base learner gains from its predecessor tree or model and updates the remaining error, as diagrammatically explained below, XGBoost Operation can be visualized as from fig.4.6.

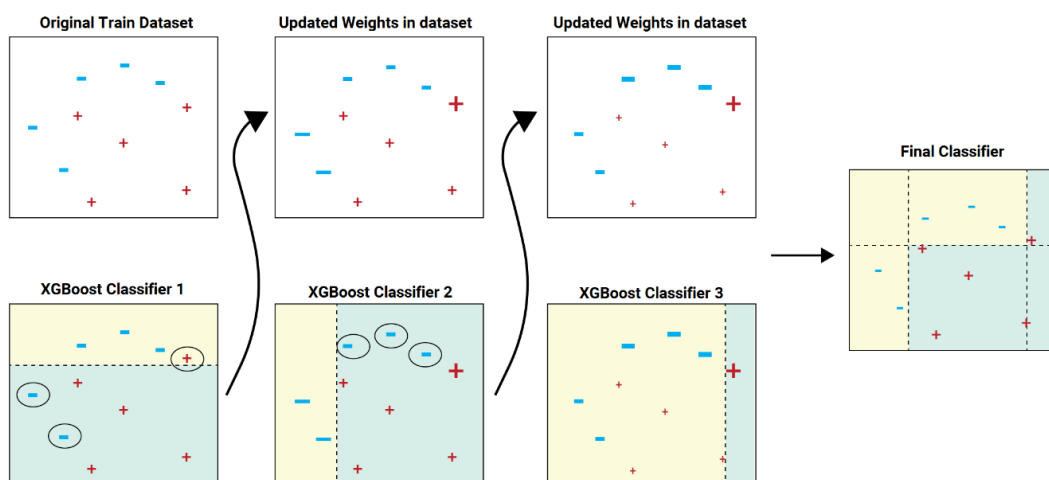


Fig. 4.6 Working of XGBOOST [24]

Consequently, the base decision model that consecutively has the motive to develop next in the arrangement gains from a modified rendition of the error. The XGBoost algorithm to be generally used avoid problems of over fitting [25]. Mathematical intuition behind gradient boosting technique is as follows:

Firstly, the boosting model is initialized to the objective function, $F_0(x)$, using eq. 4.2

$$F_0(x) = \min_{\gamma} \sum_{i=1}^n L(y_i, \gamma) \quad (4.2)$$

Then ‘m’ base learners are generated iteratively. Calculate the pseudo-residual, along with the iterative computation of gradient value of assigned loss function using eq.4.3

$$r_{im} = -\alpha \left[\frac{\partial(L(y_i, F(x_i)))}{\partial F(x_i)} \right]_{F(x)=F_{m-1}(x)} \quad (4.3)$$

‘ α ’ is the learning rate.

Then, through the properties of base learners, estimate the optimal value, which is derived for each terminal node by equation 4.4,

$$\gamma_m = \min_{\gamma} \sum L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)) \quad (4.4)$$

Where, $h_m(x_i)$ is a base decision model that tries to decrease the superimposed errors updated from the prior step.

Based on the previous tree the next base learner $F_m(x)$ is given as follows:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x) \quad (4.5)$$

Gradient Boosting technique fits the negative gradient for the optimization function in various iterations. XGBoost implements the base learners for optimizing the loss function iteratively, and instead of using linear search method to produce base learners, it takes the first derivative and the second derivative of the loss function. The XGBoost algorithm usually enhances its performance due to regularization terms included in its objective function.

4.4 LONG SHORT-TERM MEMORY (LSTM) NETWORK

With RNN, it is difficult to hold information for a longer time due to the problems of vanishing and exploding gradient hence as a solution, Long Short-Term Memory (LSTM) network were used [26]. LSTM is an improved case of RNN that have been found to have the ability to get familiar with the long-term dependencies [27]. Structurally, LSTM network, as shown in fig. 4, comprises of special units or gates which control as to when an information should enter the memory, when the information be returned, and when it is to be forgotten.

The Prime idea of LSTM network is the cell state, and other subpart gates. The cell states is conceptually tries its best to behave as a medium which consecutively, architecturally trying to transfer the incoming previous state information right through sequentially. The cell state in principle can also be considered the “memory” of the LSTM network.

Theoretically, it can convey pertinent data all through the sequence effectively and efficiently with less hindrance. Hence, the incoming sequential data or information from the previous instant of the process can advance towards latter time instances. Hence conceptually for this cell state to function and for the information to get added or removed properly, neural networks known as Gates come into picture. They basically decide on which information is to allow in a cell state. LSTM cell has three types of gates that direct the data streams, mentioned below.

Forget Gate: As the name suggests, this gate chooses what data ought to be kept discarded. Information from the past hidden state and information current input is passed through the sigmoid activation function. A sigmoid activation function is like the ‘tanh’ activation function. Rather than crunching values within -1 and 1, it crunches the values to make them within the range of 0 and 1 enabling updating or negating information as here when a value gets multiplied by 0 returns 0, leading the data points to vanish or be “neglected.” And when the value is multiplied by 1 it returns exact value meaning that value is “kept.”

Input Gate: This gate is utilized effectively to update the cell state. To start with, the information from the prior hidden layer neuron and current time input layer neuron is put into the sigmoid activation function, which accurately decides what values shall be defined for updating by the value 0 meaning not significant, and 1 meaning significant. Similarly the previous stage information is also passed into the tanh function to make the value within the range of -1 and 1. Then output from the ‘tanh’ activation function is multiplied with the output from the sigmoid activation function. Finally, the sigmoid output will choose which data is critical to keep, yielded from the ‘tanh’ activation function.

Output Gate: here the gate into close consideration, decides upon next hidden state. Since, the hidden state holds information of the past inputs, it is used for predictions. Here, again the information from past hidden state and the current input is passed into a sigmoid function. After which recently adjusted cell state is passed to the tanh function. This output from the ‘tanh’ function is multiplied with the output from the sigmoid activation function to choose the information that the hidden state must convey further. Following the above step, outcome given is the new hidden state. Both states that are, new cell state and the new hidden state are then transferred through to the upcoming time instant as in fig 4.7.

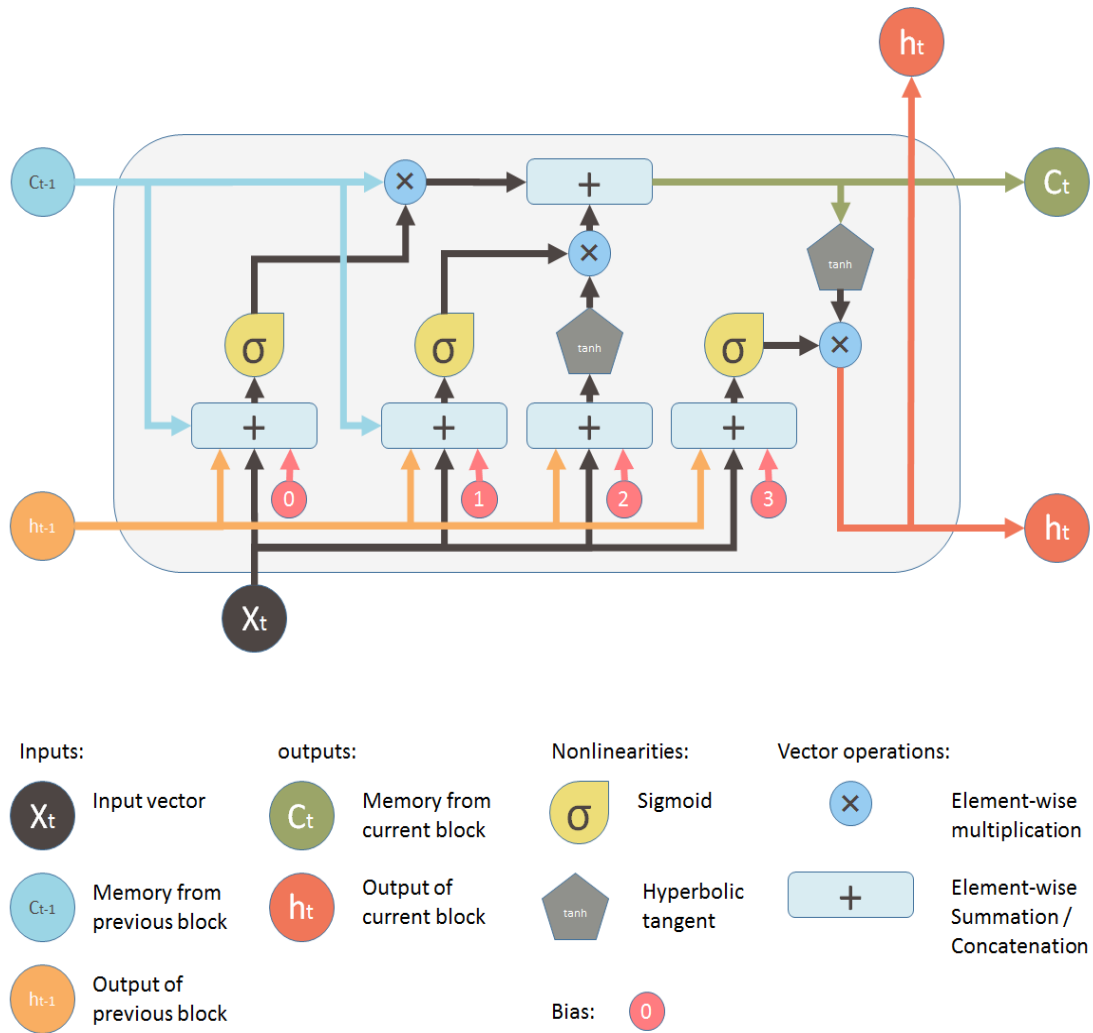


Fig. 4.7 working of a LSTM unit [28]

The working of an LSTM network can be explained mathematically as follows. Firstly, the forget gate informs which information is to be considered and which is to be discarded using current input value depicted by $X(t)$ and prior hidden state depicted by $h(t-1)$ and the sigmoid function, which gives values which are within 0 and 1 signifying absolutely keep it, and absolutely forget it, respectively. The same is represented mathematically by eq. 4.6.

$$f_t = \sigma(W_f) * (h_{t-1}, x_t) + b_f \quad (4.6)$$

Here $\sigma(\cdot)$ = sigmoid function.

Eqs. 4.7 to 4.12 below, represent the later stages when the state $X(t)$ and past state $h(t-1)$ are applied to the second sigmoid function. A vector ($C'(t)$) with value within -1 and 1 is created when the information from the preceding state is passed via the function 'tanh'. The output values from the activation function are point-by-point multiplied. Multiplication of the preceding cell state C_{t-1} is done with forget vector $f(t)$ and in case of 0, the values get dropped in the cell state. Next, a new cell state $C(t)$ is updated after point to point addition. Lastly, the output gate defines a value for the upcoming hidden state.

$$i_t = \sigma(W_i * (h_{t-1}, x_t) + b_i) \quad (4.7)$$

$$C'_t = \tanh(W_c * (h_{t-1}, x_t) + b_c) \quad (4.8)$$

$$C_t = f_t * C_{t-1} + i_t * C'_t \quad (4.9)$$

$$O_t = \sigma(W_o * (h_{t-1}, x_t) + b_o) \quad (4.10)$$

$$h_t = O_t * \tanh(C_t) \quad (4.11)$$

$$C'_t = \tanh(W_c * (h_{t-1}, x_t) + b_c) \quad (4.12)$$

The success of the neural networks in varied challenging prediction problems is credited to their depth, such as layering LSTM hidden layers increases the depth of the model, and all the more accurately depicting a deep learning technique [29].

CHAPTER 5

RESULTS AND DISCUSSION

5.1 EXPERIMENTAL SETUP

For the proper functioning of the models, and data processing the experimental setup comprises of the following hardware and software arrangements:

I. HARDWARE

1. Intel Core i3-5005U CPU
2. 8 GB DDR-3 RAM
3. 240 GB SSD Storage

II. SOFTWARE & SIMULATION LIBRARIES

1. Jupyter notebook
2. Python 3
3. Pandas
4. Keras module
5. NumPy
6. Matplotlib
7. Scikit-Learn

5.2 MODEL PARAMETERS

1. For each deep learning model, the sequence length/look back value used was 20, hence previous 20 record were used to predict the 21st data value.
2. **RNN MODEL:** Here three layers of simple RNN networks were used on top of each other where, each layer had 40 hidden nodes. The activation function used was ‘tanh’. After each RNN layer dropout layer was added to avoid overfitting.
3. **LSTM model 1:** For the first LSTM model, 3 layers of LSTM network were used with each layer comprising of 40 hidden nodes and ‘tanh’ as the activation function. The dropout layer with rate as 0.15.
4. **LSTM model 2:** LSTM model 1 was hyper tuned by increasing the hidden units in the first LSTM layer to 60 hidden nodes while the remaining two layers maintained 40 hidden units. The activation function used in this model was ‘tanh’ and the three dropout layers after each LSTM layer had 0.15 as rate depicting the fraction of input units to drop.
5. **XGBoost Model:** 1000 estimators were used with the learning rate of 0.3 and maximum tree depth of the base learner to be 6. Parameter reg_lambda was set to 1, signifying L2 regularization was applied.

5.3 SIMULATION RESULTS

80% of the entire dataset was utilized for training the models and 20% for testing. While training the RNN and LSTM models, there were 10 epochs utilized. Below are the training patterns for RNN model, LSTM model 1 and LSTM model 2 of the 10 epochs. Mean squared error (MSE) is the loss, we are trying to reduce at each epoch while training.

TABLE 5.1 TRAINING PATTERN FOR RNN MODEL

EPOCH	LOSS (MSE)
1/10	0.2358
2/10	0.0485
3/10	0.0201
4/10	0.0123
5/10	0.0091
6/10	0.0071
7/10	0.0059
8/10	0.0050
9/10	0.0044
10/10	0.0039

TABLE 5.2 TRAINING PATTERN FOR LSTM MODEL 1

EPOCH	LOSS (MSE)
1/10	0.0502
2/10	0.0121
3/10	0.0105
4/10	0.0055
5/10	0.0042
6/10	0.0071
7/10	0.0033
8/10	0.0023
9/10	0.0021
10/10	0.0019

TABLE 5.3 TRAINING PATTERN FOR LSTM MODEL 2

EPOCH	LOSS (MSE)
1/10	0.0546
2/10	0.0123
3/10	0.0114
4/10	0.0090
5/10	0.0048
6/10	0.0039
7/10	0.0031
8/10	0.0023
9/10	0.0021
10/10	0.0018

As observed from the above tables LSTM model 2 has tried to obtain the minimum loss of 0.0018, whereas the RNN and LSTM model 1 could reduce MSE to be 0.0039 and 0.0019.

All the 4 models are tested using 20% of data after training and their performance can be assessed graphically with plots of two-week predictions and single day hourly forecasts for each model.

5.3.1 RNN

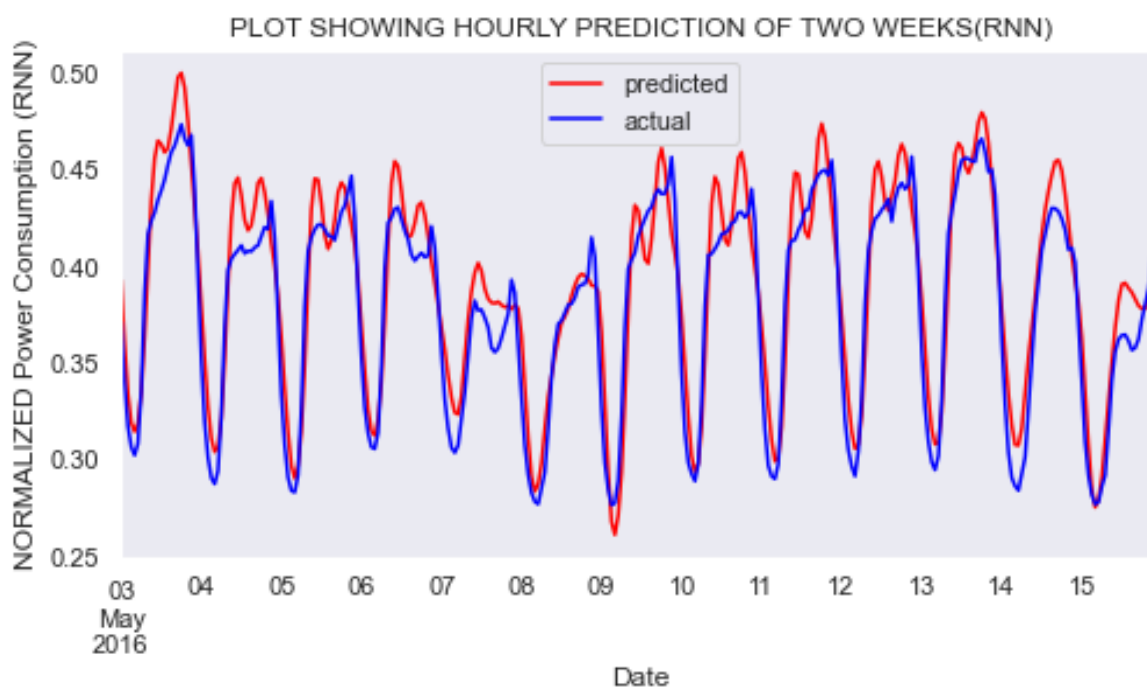


Fig. 5.1 Two Weeks prediction plot for simple RNN model

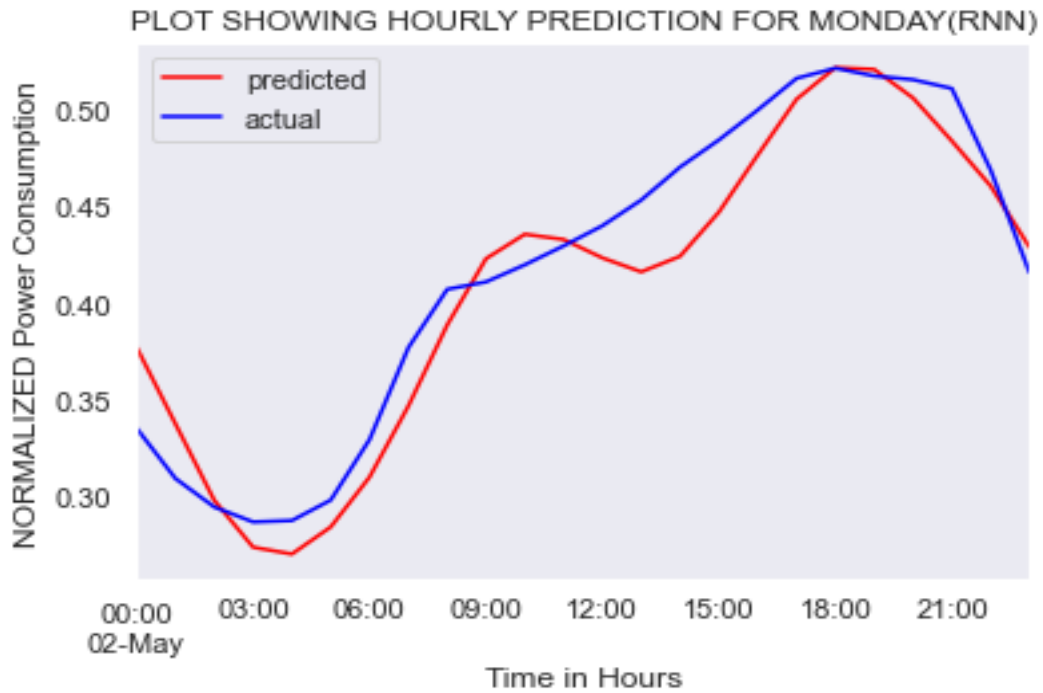


Fig. 5.2 Single day hourly forecast by RNN model

5.3.2 LSTM MODEL 1

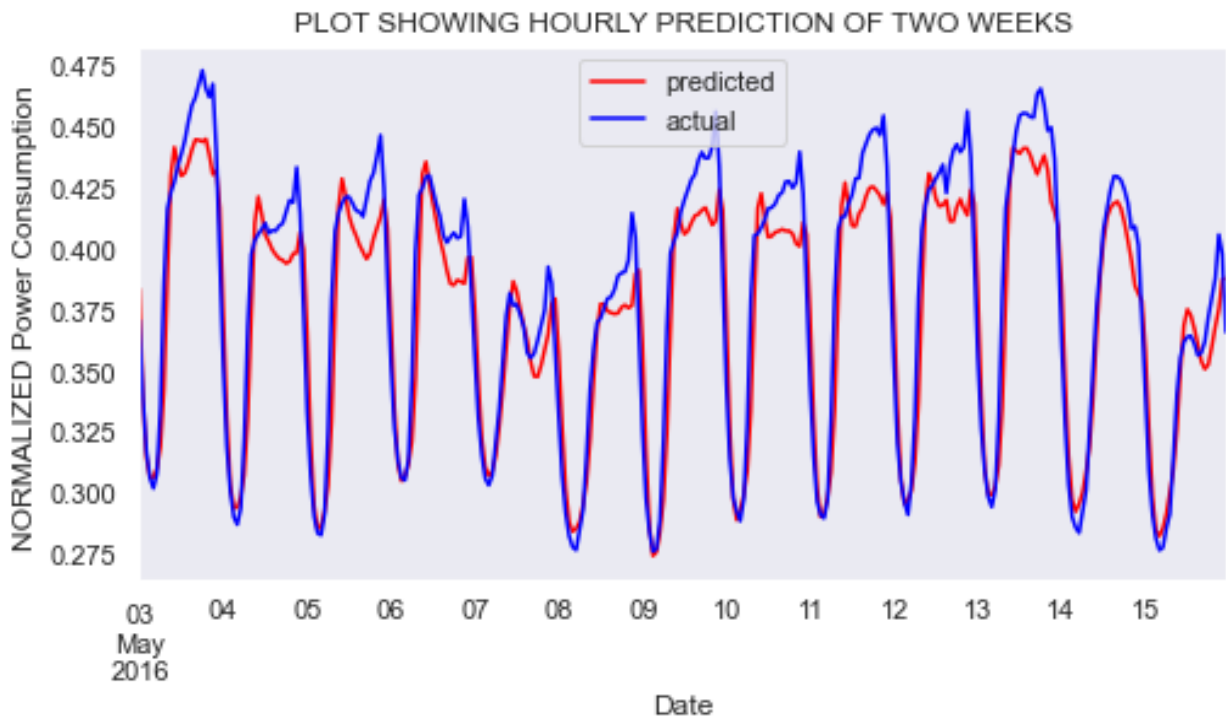


Fig. 5.3 Two Weeks prediction plot for LSTM model 1

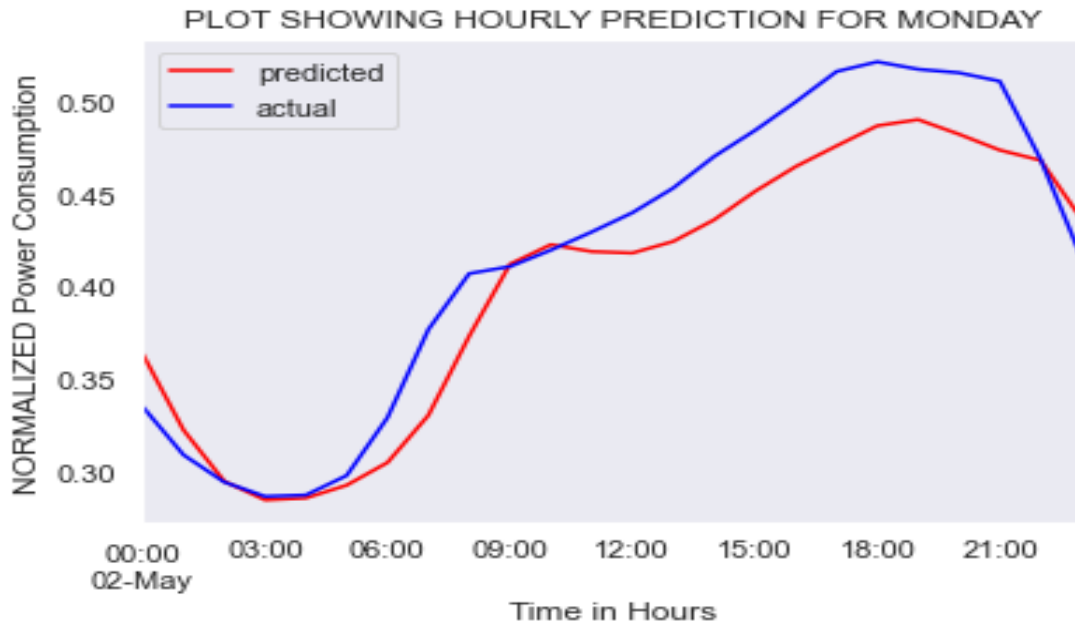


Fig. 5.4 Single day hourly forecast by LSTM model 1

5.3.3 LSTM MODEL 2

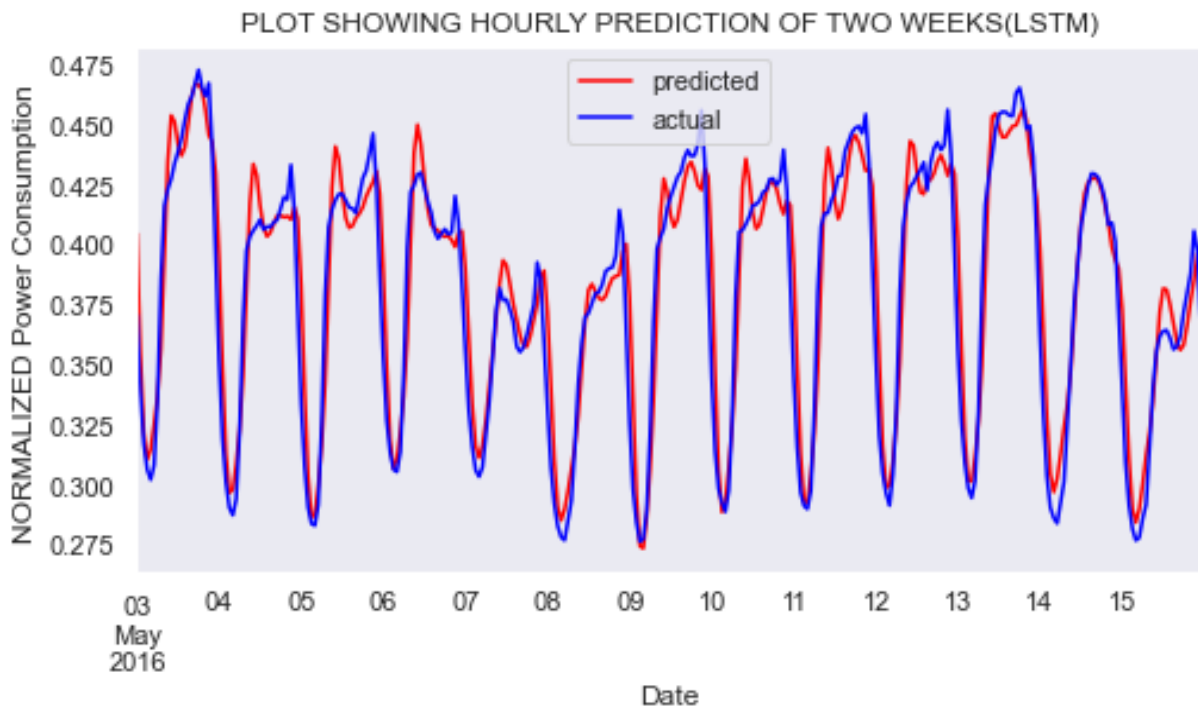


Fig. 5.5 Two Weeks prediction plot for LSTM model 2

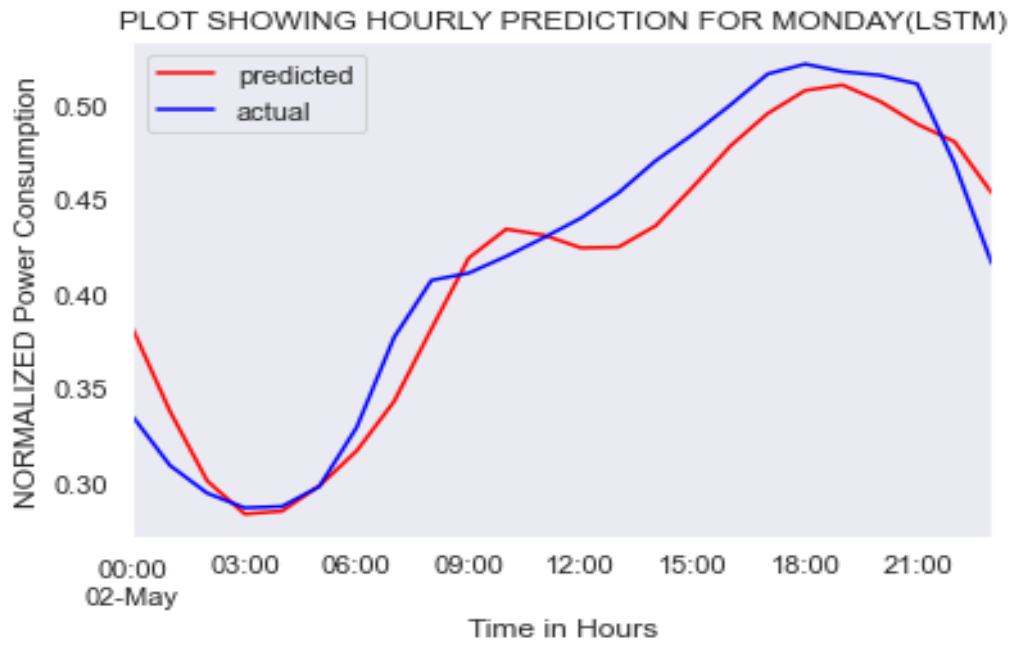


Fig. 5.6 Single day hourly forecast by LSTM model 2

5.3.4 XGBOOST

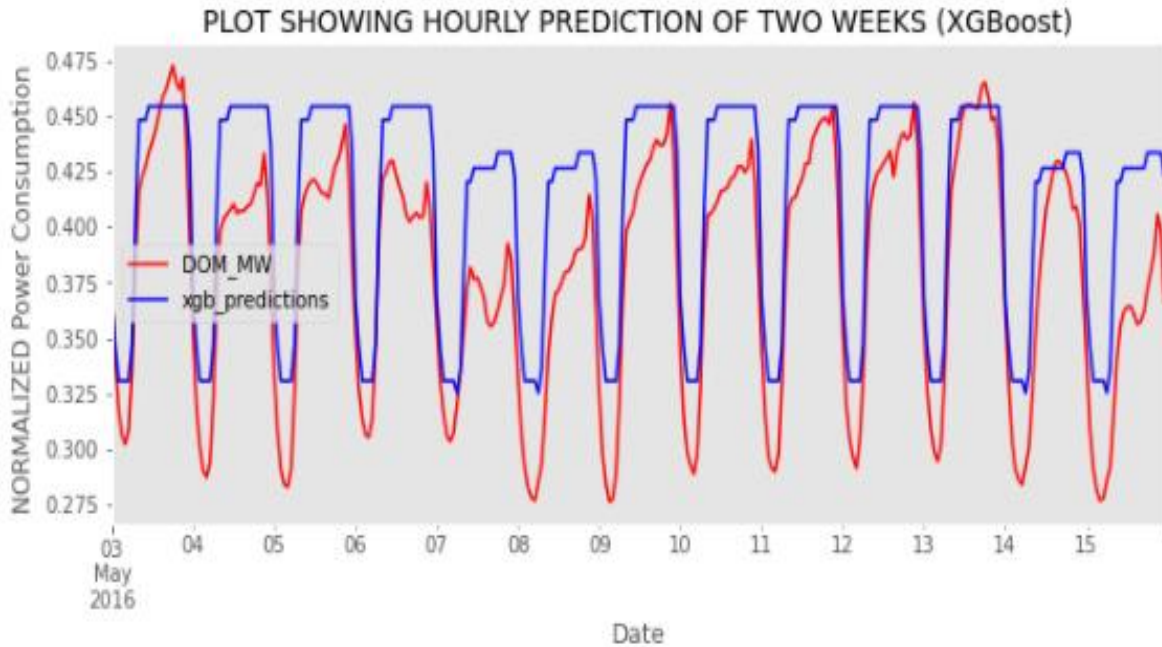


Fig. 5.7 Two Weeks prediction plot for XGBOOST model

PLOT SHOWING HOURLY PREDICTION FOR MONDAY (XGBoost)

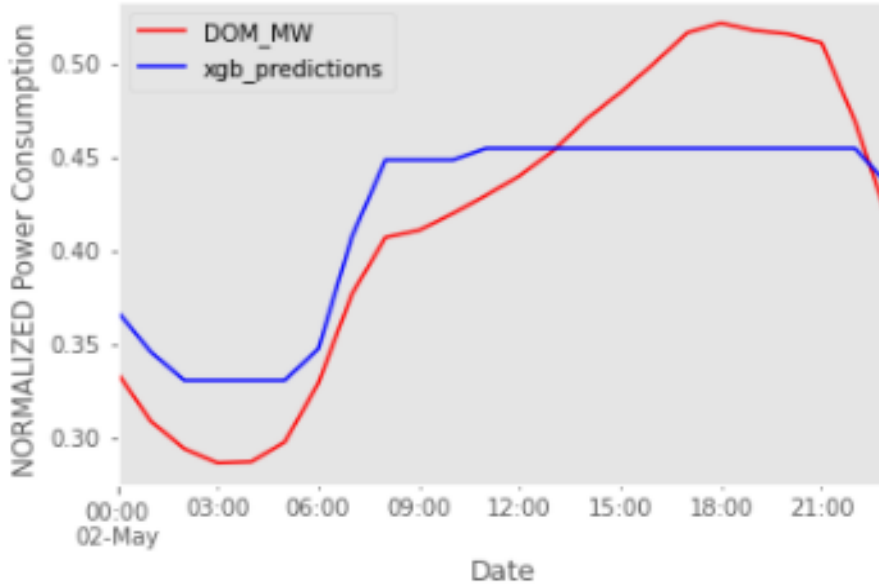


Fig. 5.8 Single day hourly forecast by XGBOOST

5.4 PERFORMANCE METRICS

The models' performance was evaluated using R2 score along with RMSE through a comparative investigation.

1. **R2 SCORE:** pronounced as R squared score, is a statistical metric that addresses the accuracy of fitted machine learning algorithm. The ideal value for the mentioned measure is 1, i.e., closer the R2 score value is to 1, better the model has been fit. This measure is a comparison of residual sum of squares with the total sum of squares. The R2 score was obtained using eq. 5.1,

$$R2 \text{ score} = 1 - \frac{\sum (T_i - F_i)^2}{\sum (T_i - \bar{F})^2} \quad (5.1)$$

2. **RMSE:** This evaluation metrics in statistics stands for root mean squared error, which is a metric which is the squared root of the mean of the square of the error. It is the measure of how well a model fits the data values. RMSE was calculated using eq.5.2

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (F_i - T_i)^2}{N}} \quad (5.2)$$

3. **MAE:** stands for mean absolute error. (MAE) is a statistical metric which measures the error between observations addressing the same feature in the data set. Hence, It is the arithmetic average of the absolute difference between the predicted and target values. It was calculated using the eq. 5.3.

$$MAE = \frac{1}{N} \sum_{i=1}^N \left| \frac{T_i - F_i}{T_i} \right| \quad (5.3)$$

In the above equations, F_i signifies the forecasted value, T_i depicts the actual value for the i^{th} instance, \bar{F} depicts the mean of the forecasted values and N depicts the total no. of instances.

TABLE 5.9 EVALUATION METRICS FOR THE USED ALGORITHMS

MODELS	R2 SCORE	RMSE	MAE
RNN MODEL	0.959	0.02446	0.04249
LSTM MODEL 1	0.953	0.02614	0.0405
LSTM MODEL 2	0.973	0.01996	0.0329
XGBOOST	0.723	0.08595	0.1321

As it is very much evident from table 5.9 that the hypertuned LSTM model i.e., the LSTM model 2, where we increased number of hidden nodes gave an improved value of R2 score which was 0.973 as compared to the LSTM model 1 having R2 score of 0.953 and the deep RNN model which had 0.959 as R2 score. XGBoost was also not able to perform well giving 0.723 as the R2 score. The RMSE and MAPE values of LSTM model 2 was also impressive in comparison to the other three models.

5.5 RELATIVE ERROR PLOTS

For investigating and evaluating the performance of the models, the plots showing the relative error values for each data point for the 4 models has been shown below.

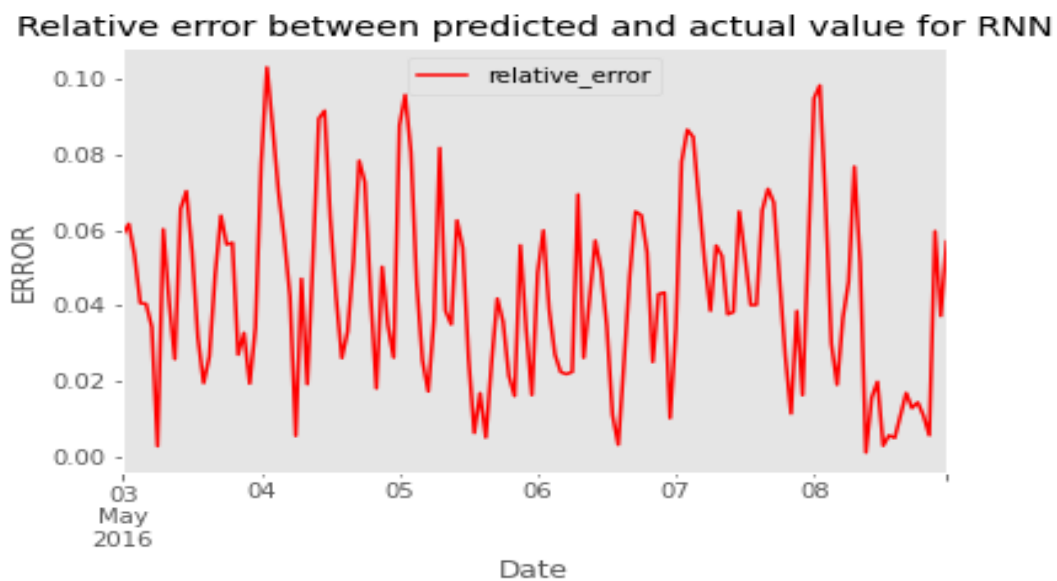


Fig. 5.9 Plot of Relative error for RNN model predictions

Relative error between predicted and actual value for LSTM

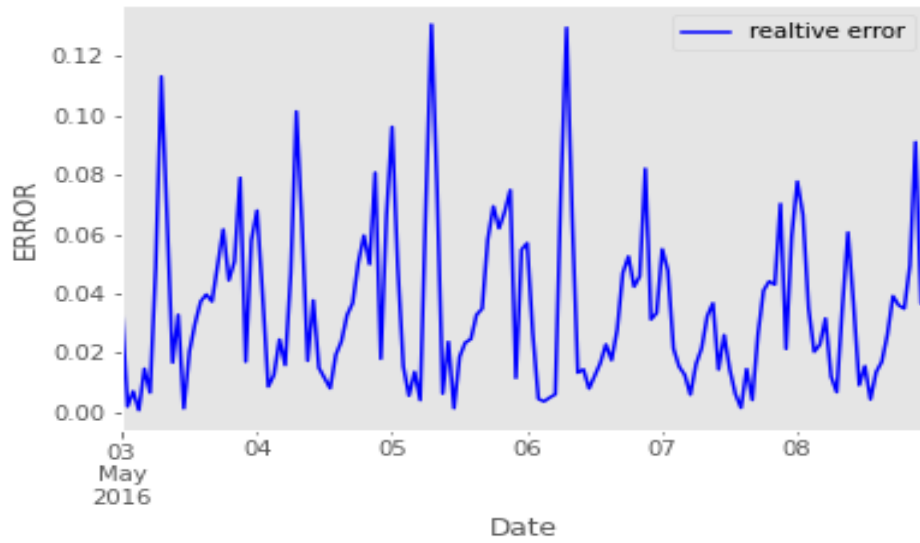


Fig. 5.10 Plot of Relative error for LSTM model 1 predictions

Relative error between predicted and actual value for IMPROVED LSTM

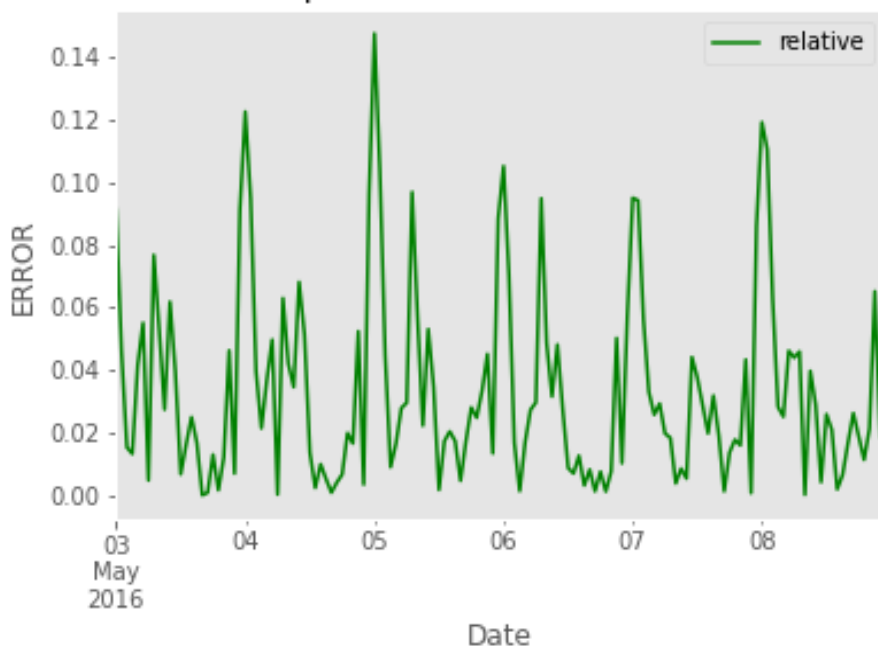


Fig. 5.11 Plot of Relative error for LSTM model 2 predictions

Relative error between predicted and actual value for XGBOOST



Fig. 5.12 Plot of Relative error for XGBoost model predictions

As clearly can be observed from the plots above, for the LSTM Model 2 (fig. 5.11), more concentration of error values is between 0 and 0.07 hence the mean error is significantly decreased as compared to RNN where a greater number of relative error values lie within 0.02 and 0.08, leading to a greater mean value of relative error (fig. 5.9). And XGBoost, gave the error between 0.05 to 0.16 (fig. 5.12).

CHAPTER 6

CONCLUSION & FUTURE SCOPE

The proposed work initialized the aim to evaluate the efficiency for short term electricity demand forecast by XGBoost, stacked LSTM and RNN. Above sections of the previous chapters present the results found out by predictive analysis using boosting machine learning algorithm XGBoost, Deep learning algorithms LSTM and RNN. It also sheds light on the normalization of data. This study extracted the results after hypertuning the LSTM model 1 to give LSTM model 2, to check if there was any significant improvement in accuracy of the predictions.

The algorithms used for predictions are an advanced clan of algorithms where XGBoost belongs to ensemble boosting categorization and LSTM is able to negate exploding gradient of RNN and can handle long term dependencies. Hence, upon predictive analysis it was proved that stacked LSTM with 60 hidden nodes in its 1st layer performed way better than XGBoost that used 1000 estimators at the backend and even better than deep RNN with 40 hidden nodes. As shown before, the stacked LSTM had the lowest MAE and RMSE values of 0.0329, and 0.01996 respectively, compared to the other two models. Hence it can be said that for a dataset such as that from the PJM energy market, stacked LSTM with the above hyperparameters can be used as a baseline model for accurate predictions.

The current work can be expanded, to utilize hybrid models combining the boosting algorithms for the purpose of feature importance and LSTM model for final predictive analysis. In the current study univariate time series prediction was done for LSTM and RNN models. In future, more independent feature can be included for multivariate electricity load prediction using deep learning algorithms. These features can be weather details such as temperature, solar exposure, humidity, Electricity price, etc.

REFERENCES

- [1] Factors that affect electrical power demand - <https://engineering.electrical-equipment.org/electrical-distribution/electric-load-forecasting-advantages-challenges.html>.
- [2] Elkarmi, F., & Abu Shikhah, N., "Power System Analysis and Studies: Cooperation between Academia and Electricity Companies", CIGRE Conference, Amman, Jordan, 2007.
- [3] Khaled, M., EL-Naggar, K.M. & AL-Rumaih K.A, "Electric Load Forecasting Using Genetic Based Algorithm, Optimal Filter Estimator and Least Error Squares Technique: Comparative Study", Proceedings Of World Academy Of Science, Engineering And Technology (PWASET), 2005, 6, pp. 138-142.
- [4] Zhou Kaile et al. "A review of electric load classification in smart grid environment", Renewable and sustainable energy reviews, 2013, Vol 24, pp 103-110.
- [5] Grzegorz Dudek, "Pattern-based local linear regression models for short-term load forecasting". Electric Power Systems Research, 2016, pp 139–147.
- [6] Saahil Shenoy, Dimitry Gorinevsky, Stephen Boyd, "Non-parametric regression modeling for stochastic optimization of power grid load forecast", In Proceedings of the American Control Conference (ACC), Chicago, IL, USA, 1–3 July 2015, pp. 1010–1015.
- [7] S. Shastri, A. Sharma, V. Mansotra, A. Sharma, A. Bhadwal, M. Kumari, "A Study on Exponential Smoothing Method for Forecasting", International Journal of Computer Sciences and Engineering 6, 2018, pp. 482-485.
- [8] A. Ghanbari, S.M.R. Kazemi, F. Mehmanpazir, M. M. Nakhostin, "A Cooperative Ant Colony Optimization-Genetic Algorithm approach for construction of energy demand forecasting knowledge-based expert systems", Knowledge-Based Systems, Vol 39, 2013, pp. 194-206.
- [9] L. Suganthi, S. Iniyar, A.A. Samuel, "Applications of fuzzy logic in renewable energy systems—A review", Renew. Sustain. Energy Rev. 2015, 48, pp 585–607.
- [10] Selakov, A.; Cvijetinović, D.; Milović, L.; Mellon, S.; Bekut, D. "Hybrid pso-svm method for short-term load forecasting during periods with significant temperature variations in city of Burbank" Appl. Soft Comput. 2014, 16, 80–88.
- [11] S. Tzafestas and E. Tzafestas, "Computational intelligence techniques for short term electric load forecasting", Jour. of intelligent and robotic sys., 2001, Vol 31, pp 7–68.
- [12] S Rahman, O Hazim, "A generalized knowledge based short term load forecasting technique", IEEE Trans. on power sys., 1993, Vol 8, Issue 2, pp 508-514.

- [13] Ankit Kumar Srivastava, "Short term load forecasting using regression trees, random forest, bagging and M5P", *Int. journal of adv. trends in computer science and engg.*, 2020, pp 1898-1902.
- [14] Dahl M et al., "Improving short term heat load forecasts with calendar and holiday data", *Energies*, 2018, Vol 11, pp 1678-1682.
- [15] Zakarya S, Abbas H, Belal M, "Long-term deep learning load forecasting based on social and economic factors in the Kuwait region", *Journal of theoretical and applied information tech.*, 2017, pp 1524-1535.
- [16] Q W Luthuli, K A Folly, "Short term load forecasting using artificial intelligence", *IEEE PES power Africa, Zambia*, 2016, pp 129-133
- [17] L. Wu and M. Shahidehpour, "A hybrid model for integrated day-ahead electricity price and load forecasting in smart grid," no. April, pp. 1937–1950, 2014], ARIMA was combined with SVM and Neural Networks.
- [18] Quilumba F et al., "Using smart meter data to improve the accuracy of intraday load forecasting considering customer behaviour similarities", *IEEE Trans. on smart grid*, 2015, Vol 6, pp 911-918.
- [19] X. Liao, N. Cao, M. Li and X. Kang, "Research on Short-Term Load Forecasting Using XGBoost Based on Similar Days," 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2019, pp. 675-678, doi: 10.1109/ICITBS.2019.00167.
- [20] M. A. Yahya, S. P. Hadi and L. M. Putranto, "Short-Term Electric Load Forecasting Using Recurrent Neural Network (Study Case of Load Forecasting in Central Java and Special Region of Yogyakarta)," 2018 4th International Conference on Science and Technology (ICST), 2018, pp. 1-6.
- [21] Gers, F.A.; Schmidhuber, J.; Cummins, F, "Learning to forget: Continual prediction with lstm", *Neural Comput.* 2000, 12, 2451–2471.
- [22] Hourly-energy-consumption - <https://www.kaggle.com/robikscube/hourly-energy-consumption>.
- [23] T Jayalakshmi, A Santhakumaran, "Statistical normalization and back propagation for classification", *Int. Journal of computer theory and engg.*, 2011, Vol 3, Issue 1, pp 1793-1801.
- [24] Xgboost-python - <https://blog.quantinsti.com/xgboost-python/>
- [25] M. Al-Rakhami, A. Gumaei, A. Alsanad, A. Alamri and M. M. Hassan, "An Ensemble Learning Approach for Accurate Energy Load Prediction in Residential Buildings," in *IEEE Access*, vol. 7, pp. 48328-48338, 2019.
- [26] S. S. Namini, N. Tavakoli, A. S. Namin,"A Comparison of ARIMA and LSTM in Forecasting Time Series", 2018 pp. 1394-1401.

- [27] Y Bengio, P Simard, P Frasconi, "Learning long term dependencies with gradient descent is difficult," IEEE Trans. on neural networks, 1994, Vol 5, Issue 2, pp 157-166.
- [28] Understanding-LSTM-and-its-diagrams-<https://blog.mlreview.com/understanding-lstm-and-its-diagrams-37e2f46f1714>.
- [29] Hermans M. and Schrauwen B, "Advances in neural information processing systems", Curran Associates Inc., NIPS 2013, Vol 26.

APPENDIX-I

PYTHON CODE

- **Data preprocessing, training and testing using XGBoost**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import r2_score

data = pd.read_csv(r'C:\0_data\data sets\archive\DOM_hourly.csv')

data.sort_values(by="Datetime", inplace=True)

data["Datetime"] = pd.to_datetime(data["Datetime"])

data.reset_index(inplace=True)

df = data.set_index("Datetime")

# normalizing the data
import sklearn.preprocessing
def normalize_data(initialData2):
    scaler = sklearn.preprocessing.MinMaxScaler()

    initialData2['DOM_MW']=scaler.fit_transform(initialData2['DOM_MW'].values.reshape(-1,1))
    return initialData2

data_norm = normalize_data(df)
data_norm.shape

def create_features(df, label=None):
    """
    Creates time series features from datetime index
    """
    df['date'] = df.index
    df['hour'] = df['date'].dt.hour
    df['dayofweek'] = df['date'].dt.dayofweek
    df['quarter'] = df['date'].dt.quarter
    df['month'] = df['date'].dt.month
    df['year'] = df['date'].dt.year
    df['dayofyear'] = df['date'].dt.dayofyear
    df['dayofmonth'] = df['date'].dt.day
    df['weekofyear'] = df['date'].dt.weekofyear

    X = df[['hour', 'dayofweek', 'quarter', 'month', 'year',
            'dayofyear', 'dayofmonth', 'weekofyear']]
    if label:
        y = df[label]
        return X, y
```

```

return X

x_train, y_train = create_features(data_train, label='DOM_MW')
x_test, y_test = create_features(data_test, label='DOM_MW')

import xgboost as xgb
from xgboost import plot_importance, plot_tree
from sklearn.metrics import mean_squared_error, mean_absolute_error

xg_reg = xgb.XGBRegressor(n_estimators=1000)
xg_reg.fit(x_train, y_train,
           eval_set=[(x_train, y_train), (x_test, y_test)],
           early_stopping_rounds=50, verbose=False)

data_test['xgb_predictions'] = xg_reg.predict(x_test)

xgb_score = r2_score(y_test, xgb_predictions)
print("R^2 Score of XGBoost model = ", xgb_score)

week_2_xgb=final_data.iloc[3489:3801]
week_2_xgb=week_2_xgb.set_index(week_2_xgb["Datetime"])
week_2_xgb_copy = week_2_xgb[['DOM_MW', 'xgb_predictions']]
week_2_xgb_copy.plot(figsize=(10,4), legend=True, color = ["red", "blue"])
plt.title('PLOT SHOWING HOURLY PREDICTION OF TWO WEEKS (XGBoost)')
plt.ylabel("NORMALIZED Power Consumption ")
plt.xlabel("Date")

plt.grid(False)
plt.show()

```

- **Data preprocessing, training and testing using RNN**

```

import os
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
import sklearn.preprocessing
from sklearn.metrics import r2_score
import keras

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.layers import LSTM
import plotly.graph_objects as go

initialData=pd.read_csv(r'C:\0_data\data
sets\archive\DOM_hourly.csv')

initialData.sort_values(by="Datetime", inplace=True)
initialData=initialData.reset_index()

train_dates = pd.to_datetime(initialData['Datetime'])

```

```

date_train = initialData['Datetime'][:92951]
date_test = initialData['Datetime'][92951:]

def load_data(stock, seq_len):
    X_train = []
    y_train = []
    for i in range(seq_len, len(stock)):
        X_train.append(stock.iloc[i-seq_len : i, 0])
        y_train.append(stock.iloc[i, 0])

    #1 last 6189 days are going to be used in test
    X_test = X_train[92951:]
    y_test = y_train[92951:]

    #2 first 110000 days are going to be used in training
    X_train = X_train[:92951]
    y_train = y_train[:92951]

    #3 convert to numpy array
    X_train = np.array(X_train)
    y_train = np.array(y_train)

    X_test = np.array(X_test)
    y_test = np.array(y_test)

    #4 reshape data to input into RNN models
    X_train = np.reshape(X_train, (92951, seq_len, 1))

    X_test = np.reshape(X_test, (X_test.shape[0], seq_len, 1))

    return [X_train, y_train, X_test, y_test]

#creating train, test data splits

seq_len = 20 #choose a sequence length

X_train, y_train, X_test, y_test = load_data(data_norm, seq_len)

print('X_train.shape = ',X_train.shape)
print('y_train.shape = ', y_train.shape)
print('X_test.shape = ', X_test.shape)
print('y_test.shape = ',y_test.shape)

# building an RNN model
from tensorflow.keras.layers import SimpleRNN
from tensorflow.keras.layers import Dropout,Dense

rnn_model = Sequential()

rnn_model.add(SimpleRNN(40,activation="tanh",return_sequences=True,
input_shape=(X_train.shape[1],1)))
rnn_model.add(Dropout(0.15))

rnn_model.add(SimpleRNN(40,activation="tanh",return_sequences=True))

```

```

rnn_model.add(Dropout(0.15))

rnn_model.add(SimpleRNN(40,activation="tanh",return_sequences=False))
rnn_model.add(Dropout(0.15))

rnn_model.add(Dense(1))

rnn_model.summary()

# training an RNN model
rnn_model.compile(optimizer="adam",loss="MSE")
rnn_model.fit(X_train, y_train, epochs=10, batch_size=1000)

# testing an RNN model
rnn_predictions = rnn_model.predict(X_test)
rnn_score = r2_score(y_test, rnn_predictions )
print("R^2 Score of LSTM model = ",rnn_score)

# results plotting
hourly=result.iloc[3489:3513]
hourly2=hourly.set_index(hourly["Date"]).drop("Date",axis=1)
hourly2.plot(figsize=(6,4),legend=True,color = ["red","blue"])
plt.title('PLOT SHOWING HOURLY PREDICTION FOR TUESDAY')
plt.ylabel("NORMALIZED Power Consumption")
plt.xlabel("Time in Hours")
plt.grid(False)
plt.show()

from matplotlib import style
import seaborn as sns

fig = plt.figure()
ax1 = plt.subplot2grid((1,1), (0,0))
style.use('ggplot')

sns.lineplot(x=hourly["Date"], y=hourly["actual"], data=hourly)
sns.lineplot(x=hourly["Date"], y=hourly["predicted"], data=hourly)
sns.set(rc={'figure.figsize':(10,6)})

plt.title("PLOT SHOWING HOURLY PREDICTION OF A SINGLE DAY")
plt.xlabel("Time in Hours")
plt.ylabel("NORMALIZED Power Consumption")
plt.grid(True)
plt.legend(["actual","predicted"])

```

- **Data preprocessing, training and testing using LSTM**

```

import os
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
import sklearn.preprocessing
from sklearn.metrics import r2_score
import keras

from tensorflow.keras.models import Sequential

```



```

from tensorflow.keras.layers import Dense,Dropout
from tensorflow.keras.layers import LSTM
import plotly.graph_objects as go

initialData=pd.read_csv(r'C:\0_data\data
sets\archive\DOM_hourly.csv')
initialData.sort_values(by="Datetime",inplace=True)

initialData=initialData.reset_index()

train_dates = pd.to_datetime(initialData['Datetime'])

initialData2=initialData.copy()

initialData2.set_index(initialData2["Datetime"],inplace=True)

# visualization of data before noramlization
initialData2.plot(figsize=(16,4),legend=True,color = "green")
plt.title('DOM hourly electrcity demand - BEFORE NORMALIZATION')
plt.ylabel("Power Consumption in MW")
plt.show()

# normalizing the data
def normalize_data(initialData2):
    scaler = sklearn.preprocessing.MinMaxScaler()

initialData2['DOM_MW']=scaler.fit_transform(initialData2['DOM_MW'].va
lues.reshape(-1,1))
    return initialData2

data_norm = normalize_data(initialData2)
data_norm.shape

# visualizing the data after noramlization
data_norm.plot(figsize=(16,4),legend=True,color= "magenta")
plt.title("DOM hourly electrcity demand - AFTER NORMALIZATION")
plt.ylabel("Normalized Power Consumption in MW ");

date_train = initialData['Datetime'][:92951]
date_test = initialData['Datetime'][92951:]

def load_data(stock, seq_len):
    X_train = []
    y_train = []
    for i in range(seq_len, len(stock)):
        X_train.append(stock.iloc[i-seq_len : i, 0])
        y_train.append(stock.iloc[i, 0])

    #1 last 6189 days are going to be used in test
    X_test = X_train[92951:]
    y_test = y_train[92951:]

    #2 first 110000 days are going to be used in training
    X_train = X_train[:92951]
    y_train = y_train[:92951]

```

```

#3 convert to numpy array
X_train = np.array(X_train)
y_train = np.array(y_train)

X_test = np.array(X_test)
y_test = np.array(y_test)

#4 reshape data to input into RNN models
X_train = np.reshape(X_train, (92951, seq_len, 1))

X_test = np.reshape(X_test, (X_test.shape[0], seq_len, 1))

return [X_train, y_train, X_test, y_test]

#creating train, test data splits
seq_len = 20 #choose a sequence length
X_train, y_train, X_test, y_test = load_data(data_norm, seq_len)
print('X_train.shape = ',X_train.shape)
print('y_train.shape = ', y_train.shape)
print('X_test.shape = ', X_test.shape)
print('y_test.shape = ',y_test.shape)

# Building an lstm model
lstm_model = Sequential()

lstm_model.add(LSTM(40,activation="tanh",return_sequences=True,
input_shape=(X_train.shape[1],1)))
lstm_model.add(Dropout(0.15))

lstm_model.add(LSTM(40,activation="tanh",return_sequences=True))
lstm_model.add(Dropout(0.15))

lstm_model.add(LSTM(40,activation="tanh",return_sequences=False))
lstm_model.add(Dropout(0.15))

lstm_model.add(Dense(1))

lstm_model.summary()

# training an LSTM model
lstm_model.compile(optimizer="adam",loss="MSE")
lstm_model.fit(X_train, y_train, epochs=10, batch_size=1000)

# testing the LSTM model
lstm_predictions = lstm_model.predict(X_test)
lstm_score = r2_score(y_test, lstm_predictions)
print("R^2 Score of LSTM model = ",lstm_score)

# plotting the results
result_plot=pd.DataFrame({"Date":date_test[20:], "predicted":lstm_prediction_load, "actual":load_test})
result["Date"]=pd.to_datetime(result["Date"])
result["Weekday"]=result["Date"].dt.dayofweek

hourly=result.iloc[3489:3513]

```

```

hourly2=hourly.set_index(hourly["Date"]).drop("Date",axis=1)

hourly2.plot(figsize=(6,4),legend=True,color = ["red","blue"])
plt.title('PLOT SHOWING HOURLY PREDICTION FOR TUESDAY')
plt.ylabel("NORMALIZED Power Consumption")
plt.xlabel("Time in Hours")

plt.grid(False)
plt.show()

weeklstm2=result.iloc[3489:3801]
weeklstm2=weeklstm2.set_index(weeklstm2["Date"])
weeklstm2.drop(["Weekday","month","Date"],axis=1,inplace=True)
weeklstm2.plot(figsize=(8,4),legend=True,color = ["red","blue"])
plt.title('PLOT SHOWING HOURLY PREDICTION OF TWO WEEKS')
plt.ylabel("NORMALIZED Power Consumption ")
plt.xlabel("Date")

plt.grid(False)
plt.show()

```

APPENDIX-II

LIST OF PUBLICATION

1. "Short Term Electricity Demand Forecast Using Deep RNN And Stacked LSTM". 6th Springer, International Conference on Advanced Production and Industrial Engineering, (ICAPIE), 2021
2. "A Comparative Analysis of Extreme Gradient Boosting Technique with Long Short-Term Memory and Layered Recurrent Neural Network for Electricity Demand Forecast". 6th IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (IRTEICT), 2021.