National College of Ireland

Analysis of Electric Load Forecasts Using Machine Learning Techniques

MSc Research Project

DATA ANALYTICS

GABRIEL DADA IBUKUN

Student ID: X18176585

School of Computing

National College of Ireland

Supervisor: Dr. Vladimir Milosavljevic

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: GABRIEL DADA IBUKUN ……

Student ID: ….X18176585……..……

Programme:……DATA ANALYTICS MASTERS …… **Year:** …2019……..

Module: ……RESEARCH PROJECT……..……

Supervisor: Dr. Vladimir Milosavljevic ……

Submission Due Date: …12th DECEMBER 2020……

Project Title: **Analysis of Electric Load Forecasts Using Machine Learning Techniques**

Word Count: ……8000…… **Page Count**……28……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……

Date: ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Analysis of Electric Load Forecasts Using Machine Learning Techniques

Gabriel Dada Ibukun

X18176585

Abstract

Abstract—Load forecasting forms the basis of demand response planning in energy trading markets where smart grid operators look to achieve effective resource planning in servicing customers' electricity needs. To this end, historical data on customer electricity usage patterns have been deployed extensively in literature, applying state-of-the-art approaches in building predictive models for various use cases. In this research paper, a comparative study of different approaches for modeling time series load data is presented. The aim is to measure the impact and performance of classical statistical approaches, and more recent machine learning approaches on different lengths of historical data. To achieve this, 10 years of electric load data from PJM transmission and weather data were used to deploy day-ahead forecasts using ARIMA, SARIMA, Extra Trees regressor and XGBoost. Performance was compared using evaluation metrics such as RMSE, MAPE and MAE. Upon successful completion of experiments, results show that regression-based machine learning models generally showed better results for modeling with lengthier historical data (more than two years). This is especially so as Extra Trees regressor delivered the best RMSE of 0.425 for 6years data overall. For shorter time series length between 4 weeks to 6 months, SARIMA delivered an RMSE of 0.129, which is the best result overall.

Keywords: Electric Load Forecasting, Time Series Forecasting and Machine Learning.

1. Introduction

Demand-response forecasting is a critical component of energy management systems, as it allows for effective resource planning and distribution in modern smart grid operations through the measurement, control and optimization of load resources. As a result, over the last decade, electric load forecasting has been a subject of keen interest in academic research. During this time, several innovative approaches have been explored with the goal of continuous improvement in terms of methodologies and (or) model performance— a common denominator being the design of intelligent load forecasting systems (ILF) that help to find the balance between consumers' energy demand and the distributed resources from micro-grid energy producers.

Forecasting load inaccurately, either by overestimating or underestimating load demand hinders resource scheduling. This may result in the smart grid system operating in a vulnerable state. Conversely models that deliver accurate forecasts mitigate this risk, guaranteeing stability in power system operations. Predictive models that have been deployed in electric load forecasting (ELF) can be broadly categorized into classical statistical models and data-driven models. The classical techniques are mostly applied to effectively model historical usage patterns. This includes the use of time series models such as auto regressive integrated moving average (ARIMA) proposed by Box and Jenkins in 1994. Similarly, data-driven models have been widely deployed mostly to improve accuracy of electric load forecasting. As a result machine-learning and artificial intelligence algorithms have been leveraged to account for the intrinsic non-linear patterns in load consumption data.

The pros and cons of these approaches have been extensively studied in literature. Classical statistical models generally suffer from under-fitting and an inability to effectively model non-linear patterns. This is due to the trend and seasonal factors present in historical electric load data. As such, we can't just dump time series algorithms on such datasets for predictive analytics. While data driven models are efficient for modeling complicated non-linear relationships, they are plagued with the problem of overfitting as evident in the use of traditional neural networks for short-term load forecasting, with a high number of non-evident nodes.

Also, while electricity consumers are increasingly adopting IOT and smart home devices thereby scaling grid data volume, veracity and velocity, most state-of-the-art propositions in electric load forecasting have yet to address these changes as most models are still deployed using in-memory system on single machines. Recent breakthroughs in parallel computing can help us leverage distributed system frameworks such as spark, and machine learning algorithms (such as XGBoost) to improve processing time while dealing with large datasets. It is worthy of note that different business needs require different load forecasting models. This is because load-forecasting tasks are broadly classified into short-term (including hour-ahead, day-ahead to 2 weeks-ahead), and long term (beyond two weeks-ahead to years-ahead). As such it is imperative to understand the strength and limitations of different models giving forecast scenarios and length of data.

In this research, several approaches will be used to deal with the stochastic nature of the time series data (testing for and achieving stationarity), after which experiments are carried out to evaluate the performance of models such as ARIMA, SARIMA, XGBoost Regressor, and Extra Trees Regressor, for dah-ahead forecast using different lengths of data (including one week, two weeks, three weeks, four weeks, six months and one year). This will help us address the following research question:

RQ: To what extent can various machine-learning algorithms deployed on parallel computing framework optimize the performance of electric load forecast using different scale and length of data.

SUB RQ: What machine learning approaches are best suited to modeling different lengths of data (in terms of model accuracy and processing time).

In addressing the above research questions, our proposed framework makes the following contribution to improve the efficiency of electric load forecast:

1. The use of state-of-the-art techniques (as presented in literature) to deal with the stochastic nature of time series data in preparation for various modeling techniques.

2. An elaborate comparative analysis of the performance of five classical and machine learning approaches (including ARIMA, SARIMA, XGBoost, and Extra Trees Regressor) in electric load forecasting (ELF), vis-à-vis model accuracy and processing time on different lengths of data.

3. Ascertain the size of historical data that results in optimized forecasts for each approach.

The following sections of the paper are organized as follows. Section 2 presents the review of relevant literature. In section 3, the research methodology is described while section 4 presents the proposed design prototype. Details of all steps taken in the solutions deployment are described with corresponding outputs in section 5. Experiment results and evaluation are presented in section 6, while section 7 presents further discussion of results. Section 8 concludes the research with a suggestion on future work.

2. Literature Review

This research draws inspiration from previously published literatures in the data analytics subdomain of electric load forecasting. To address our research question, the solutions presented, and methodologies adopted in this paper are based on critical review of relevant literatures, where classical statistical and machine-learning approaches have been widely applied in the analysis of electric load data. The first subsection presents a review of bodies of work where time series techniques such as ARIMA have been applied for the analysis and prediction of load patterns. Next is the review of machine learning based approaches (either as lone or hybridized models). The pros and cons of these approaches are highlighted as the section concludes by summarizing the key findings as a basis for justifying this research.

2.1 Time Series Forecasting Using Classical Statistical Models

Several classical statistical methodologies have been proposed for time series based electric load forecast (ELF), however, the autoregressive moving average (ARMA) and its variants have been widely deployed for short term demand forecast [1]. These models are preferred in some quarters to deal with the time-dependent structures such as trends, seasonality and cyclicality inherent in time series datasets. [2] built a model to forecast monthly electric load consumption in Saudi Arabia using ARIMA. Likewise, the model was adopted to create short-term electric load forecasts in the state of Karnataka, India [3]. A comparative analysis of ARIMA and ARMA models using electric consumption data from residential buildings was presented by [4].

Other extensions have also been deployed in literature to deal with specific challenges of time series data, for example the Seasonal Autoregressive Integrated Moving Average (SARIMA)—which is robust enough to deal with the seasonal elements in time series datasets. Others are: Multiplicative Seasonal Autoregressive Integrated Moving Average (MSARIMA)—which integrates independent variables to SARIMA, as well as ARIMAX—which adds exogenous inputs to ARIMA. [5] sought to improve efficiency of load management, as such deployed MSARIMA to forecast peak electric load demand for five regions in India using weather features as independent explanatory variables. [6] proposed a model to optimize the cumulative energy consumption for an office building. This they achieved by modeling hours-ahead power demand for the said building using ARIMAX. Multiple linear regression (MLR) is another type of classical statistical technique that has been widely deployed on time series data for electric load forecasts. Using data curated from ISO New England and the Global Energy Forecasting Competition, [7] used separate subset of trained variables to deploy a family of regression models. A methodology for optimal precision was proposed by [8] comprising a number of techniques that included simple regression and multiple linear regression models. This was deployed to forecast gross energy consumption using India as a case study. Linear regression approach was used by [9] to analyse the daily and hourly electric load pattern of a retail building located in Houston, Texas. [10] used electricity consumption time series data of two years to evaluate the performance of Gaussian model and multiple linear regression models.

2.2 Forecasting with Machine Learning Models

A number of forecasting approaches based on machine learning techniques have been deployed either as lone or hybridized models. Fuzzy logic (FL), an offshoot of fuzzy set theory—first introduced by Lotfi A. Zadeh in 1965, maps inputs variables to output variables using "if-then" statements. This peculiarity makes FL algorithms suitable for exploring patterns from load consumption data whose input and output variables exhibit some non–linear patterns. [11] incorporated this peculiarity of FL algorithms by modeling non-linearly

correlated attributes between peak electric load and a number of weather features including temperature and humidity. The model successfully predicted year-ahead load achieving a 6.9% mean absolute percentage error value. A downside however is the failure of FL algorithms to effectively deal with potential cognitive misgivings inherent in non-linear patterns—this is an aspect Artificial Neural Networks (ANN) come in handy.

ANN algorithms offer greater tolerance for errors, characteristic generalization, alongside parallel processing and efficient learning abilities. As such they have been extensively deployed to a great degree of success in modeling non-linear relationships peculiar to electric load forecasts. In addition to this, the algorithms are architecturally flexible as such can be deployed in varying forms for tasks that include classification, pattern recognition, etc. These broad variants include Deep Learning (DL), Self Organizing Map (SOM), Extreme Learning Machine (ELM), and Multilayer Perceptron (MLP). [12] used MLP to model day-ahead and hour-ahead scenarios for electric load consumption. The model made use of the back propagation technique for data training. The hour-ahead and day-ahead forecasts achieved MAPE values of 2.06% and 1.40% respectively with the authors suggesting a deployment of furthers state-of-the-art architecture to address more data features in the future. A further innovative approach was deployed using MLP. Adaptive weighted method was used for the combination of three predictive components to dynamically predict k days hourly load (where K is stated to be a number of days between 2 to 7 days). First, the authors split the relationship between electric load and temperature into three varying sets of weekly, daily and hourly. Results show a MAPE of 2.34% and 1.67% MAPE for the daily and hourly forecasts respectively [13]. The shortcoming of this algorithm is that large MLP structures are required for training the electric load datasets. Also there is a potential risk of increased redundancy as a result of the three-pronged input components. [14] proposed a methodology to address these problems. In a bid to optimize for the overfitting challenges of large-scale data, the authors used electricity consumption time series data from a residential building to deploy a model using MLP based STFL. The model showed relatively improved accuracy for several loads. For future work, the authors suggested a deployment of the model on a parallel computing framework—as an alternate solution to handling the complex MLP algorithms. The SOM is an unsupervised clustering technique able to mirror local representation of data inputs. [15] employed a two-stage approach to improve forecast accuracy. SOM was first used to create several profiles from time series electric load data using clustering. In the second stage, MLP was then used to build the forecast model as supervised learning. This approach resulted in improved accuracy compared to the initial deployment where clustering was not applied. DL uses a combination of multiple processing layers in a neural network to analyse non-linear relationships between predictor and dependent variables. However, it is plagued with the risk of overfitting in demand side electric load forecasts. This is due to the high number of input layers. To address this issue, the literature [16] proposed a method to increase the data volume and variety fed into a pooling-based RNN. Testing on 920 smart meter customers in Ireland, the method showed impressive results. Specifically, the model achieved this by establishing correlation amongst several households thus generating more learning layers before overfitting. Also the pooling of customer usage profiles was a major factor in improving accuracy. ELM algorithms are characterized by their exceptional capacity

for generalization. As a result they are considered more efficient than conventional neural networks (CNN)—particularly for faster processing times training large datasets. An RELM (basically an ELM incorporated into RNN) was used by [17] to reduce load forecast errors in their work. However, it is worthy of mention that neural network based algorithms generally suffer from limitations such as overfitting as well as network nodes complexities that may lead to over estimation of models, amongst others. SVM is preferred in some quarters for its features that minimize risks thereby addressing the challenges of overfitting peculiar to ANN. SVM was used to build a model for short-term load demand using data from the electricity market operator in Australia [18]. The model's performance was benchmarked against that of ARIMA and MARS. The SVM model performed better, with the day-ahead forecast reaching the highest willmot's index of 0.890.

A number of machine learning models have also been employed in literature, with the aim of consolidating the strengths of different algorithms to improve forecast accuracy. Kennedy and Eberhart introduced the particle swam optimization (PSO) in 1965. The metaheuristic algorithm finds the optimal solution by initiating multiple solutions to move through multidimensional search space in a pattern that mirrors the characteristic movement of bird flocks. The algorithms require no assumptions of the situation being optimized, as such are a great fit to compliment other algorithms in hybrid architecture. For instance, PSO can be deployed with ANN to ascertain the optimal arrangement of the network, and optimize weights of the neurons. A PSO trained MLP for day-ahead load forecast was presented by [19]. The weights of the neurons were trimmed using PSO. The model achieved an accuracy surpassing that of back-propagation techniques. [20] deployed a PSO-ELM hybrid for a high fluctuation, low capacity micro grid. PSO was used to decompose and filter the data in the pre-processing phase, after which the model was trained using ELM. [21] leveraged the ability of the SVM-PSO hybrid to retain identified solutions from particles searches. In their implementation, PSO was used for feature selection while using the support vector regression for modeling. The results presented surpassed the hybrid of genetic algorithms GA and SVR.

Time series data can effectively be split into low and high frequency components using wavelet transformation (WT) making the algorithm effective for analysing signals that are dynamic. A hybrid of WT and ANN was proposed by [22]. WT was first used to split electric load into a number of frequency units. ANN algorithm was then applied to model the resulting normalized data. A test on ISO datasets showed an overall improved performance.

While a number of these models have proven effective, their downsides have been extensively highlighted. Also for different use cases, the impact of varying data lengths in ascertaining optimal performance for state-of-the-art modeling approaches have not been adequately covered.

3. Research Methodology

In this section, an overview of the applied methods adopted in addressing our research question is presented. The methods presented can be classified into two broad categories namely: statistical based approach for modeling timer series (SARIMA and ARIMA), and machine learning approach (XGBoost, Extra Trees, and Adaboost). A process flow is presented detailing the steps taken to perform the experiments, as well as a detailed overview of all algorithms employed.

3.1 Data Gathering and Requirements

This research will make use of two categories of time series datasets namely electric load data and weather data as presented in Table 1 below:

- **Load Data**, which contains 10 years of hourly energy consumption (in megawatts) for the state of North Carolina US, was sourced from the PJM utility database. PJM is a regional transmission organisation controlling the distribution of power to 7+ states in the US. The data contains time series features thereby exhibiting annual, weekly and hourly cycles.

- **Weather Data**: A strong positive correlation between temperature and electricity consumption during winter was established [23]. For this research, an OpenWeatherMap dataset was sourced from Kaggle. The datasets are time series spanning five years of weather for temperature, humidity, pressure, wind direction and wind speed for North American cities.

	Category	Data Source	Description	Features	Target Variables
1	Weather Data	Kaggle/ OpenWeatherMap (5 years of hourly weather measures)	Meteorological data containing weather measurements for several North American cities curated from OpenWeatherMap	Time Stamps Temperature, Pressure, Humidity, Wind Speed, Wind Direction	Temperature (K), Wind speed (Km/h), Relative Humidity (%) Atmospheric Pressure (milibar)
2	Load Data	PJM Power Utility Company	10 years of hourly load consumption data from PJM. The data is open source and available through the firm's website.	(Time, Day, Month, Year), and Energy consumption in Megawatts	Load Consumption (MW)

Table 1: Description of data sources

3.2 Process Flow

The process in Figure 1 below captures the steps and procedures taken in implementing the solutions of this research. The methodology adopted in this research is adapted from the knowledge discovery in data mining (KDD) framework.

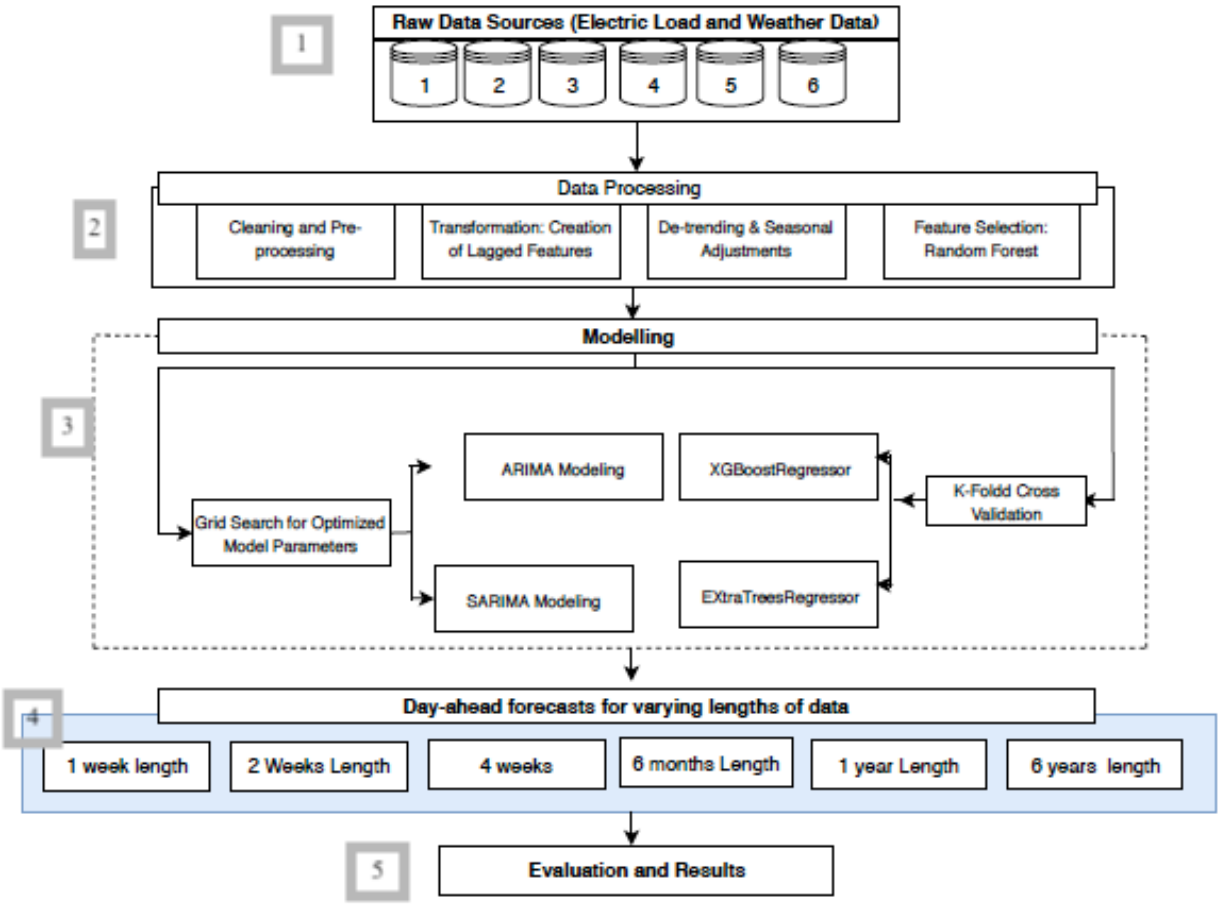

Figure 1: Implementation framework

Layer 1: Data Sources (Electric load and Weather data). Layer 2: Data Processing comprising of Cleaning and pre-processing, transformation, de-trending and seasonal adjustment. Layer 3: Modeling showing ARIMA, SARIMA, XGBoost and Extra Trees Regressor. Layer 4: Day-ahead forecasts for varying lengths of data. 5: Evaluation and Results.

3.2 Overview of Data Mining Algorithms and Techniques

The following section describes the statistical and data mining techniques and algorithms used in this research, with the aim of providing justification for their selection in addressing our research questions.

(i) **ARIMA:** Autoregressive integrated moving average (an integrated extension of the general ARMA model) was implemented using the Box-Jenkins procedure [24] recommended for time series forecasting. This method was used due to its compatibility with non-stationary time series. However a primary requirement of the model is stationarity of the time series data [25]. As such necessary inspection and adjustment for stationarity was carried out as further detailed in section 5 of this paper. A non seasonal ARIMA is defined by three parameters written as ARIMA (p, d. q) where **P is the** number of autoregressive terms (AR), **d** is the measure of differencing required to make the time series stationary (also known as the degree of differencing), and **q** is the size of the moving average window or order of moving average (MA). Using these model parameters, the ARIMA (p, d, q) forecasting equation is a linear model in the form:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \ldots + \beta_p Y_{t-p} \in_t + \phi_1 \in_{t-1} + \phi_2 \in_{t-2} + \ldots + \phi_q \in_{t-q} \ldots \quad (1)$$

Predicted (Y_t) = Constant + Linear combination lags of Y (up to p lags) + Linear Combination of lagged forecast errors (up to q lags)

To execute the ARIMA model, the first objective will therefore be to identify the parameter values p, d and q.

(ii) **SARIMA:** The Seasonal Autoregressive Integrated Moving Average provides an additional modeling of temporal influence specifically addressing potential seasonal elements in the time series. As such it is an enhancement of the ARIMA model with additional P, D, Q parameters used to model inherent seasonality in the data. It is denoted as SARIMA (p, d, q) (P, D, Q) m, where "m" signifies the periodic trend of the time series [26]. This model was selected to due to it's adaptation to seasonality especially as achieving stationarity is a necessary requirement for building forecast models on time series using classical statistical techniques. Therefor a SARIMA (p, d, q) × (P, D, Q) can be expressed as follows:

$$\varnothing(B)\,\Phi(B^S)(1-B)^d\,(1-B^s)^D\,Y_t = \theta(B)\,\Theta(Bs)\,Z_t \ldots\ldots\ldots\ldots \quad (2)$$

Where: $\phi, \theta, \Phi, \Theta$ are unknown parameters, d and D are finite non-seasonal and seasonal differencing respectively.

(iii) **Grid Search:** The process of selecting and tuning the ARIMA & SARIMA hyperparameters (p. d, q) and (P, D, Q) to achieve the best combination for modeling was achieved using the grid search procedure for a one-step rolling forecast. In python, the pandas, scikit-learn, and statsmodels libraries were used to implement an iterative model that achieves optimal parameters for our models. Essentially the process involves several iterations that studies varying combinations of hyperparameters. The parameters with the lowest AIC* score were selected for building the forecast models as suggested by [31].

(iv) **XGBoost:** The XGBoost algorithms use the principle of gradient boosting to build ensemble of trees. The algorithm has grown popular in application to electric load forecasting problems in recent years. This is mostly due to scalability as it achieves 10x speed compared to similar algorithms [27]. Also, it is able to overcome the problem to overfitting compared to Generalized Boosted Regression Models (GBM) by regularizing model formalization thereby attaining better results. Lastly, the use of out-of-core computation makes it efficiently process large-scale data even on a single machine. These qualities particularly justify the selection of XGBoost in addressing a part of our research question. Class xgb. *XGBRegressor ()* implementation provided in the scikit-learn library in python was used.

(v) **Extra Trees Regression:** The Extra trees Regressor model builds multiple randomized decision trees on several data sub-samples while using averaging to mitigate overfitting and optimize accuracy. Unlike Random forests, observations are not bootstrapped, while nodes are split randomly among feature subsets. As such they generally achieve lower variance than random forests. The class *ExtraTreeRegressor ()* implementation provided in the scikit-learn library in python was used.

(vi) **Lagging Features:** To reframe a time series problem into a supervised learning one, such that machine learning regression models can be applied effectively for forecasts, lagging features are required. In the feature engineering stage of this research, lag features of each observation was created using procedures outlined in literature [31], after which we were able to effectively apply Xgboost, Extra trees Regressor for modeling to great effects.

(vii) **Cross Validation:** K-fold was used for validation as suggested by [32]. The procedure involves randomly splitting the data into approximately equal K number of folds. One hold out fold is used as the testing set, while the aggregate of the other folds as the training set. The accuracy of the test set is calculated, and the procedure iterated K number of times reporting the average accuracy. This method makes more efficient use of data as every observation gets chance to be used featured in the training and testing, leading to more accurate results. The *cross-val-score* function in python's sklearn library was used to achieve this while setting k= 10 folds as recommended in the python sklearn manual[2].

(viii) Differencing: This procedure was used during the transformation stage to achieve stationarity of time series data as one of the assumptions of ARIMA and SARIMA modeling. To remove seasonality and trends, the difference between each observation is computed according to the equation below:

$$\boldsymbol{Y_t} = \boldsymbol{Y_t} - \boldsymbol{Y}_{t-1} \qquad (3)$$

4. Design Specification

For the purpose of reusability, the research solution is implemented via three-layer application architecture. This is presented in the figure below:

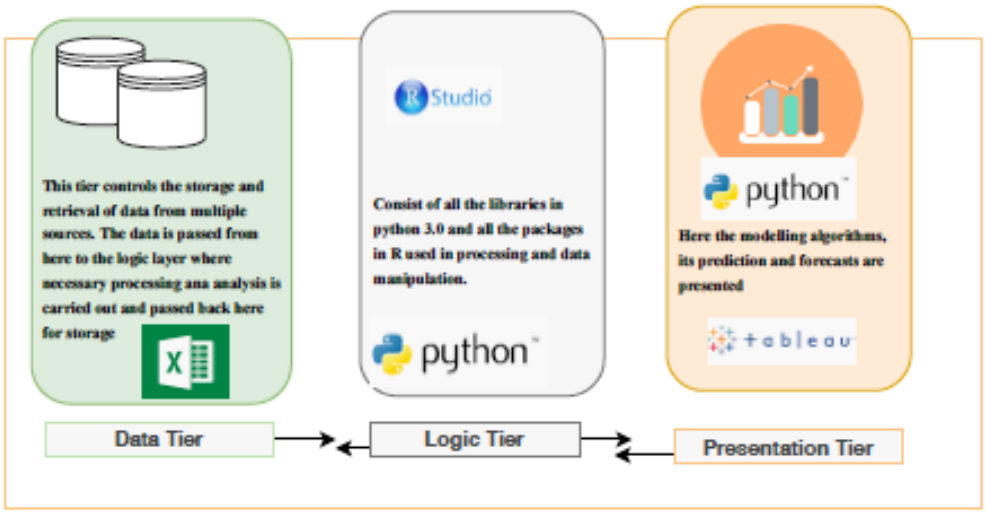

Figure 2: A Three-tier Design Architecture

1: Data Tier for storage and retrieval of data from multiple sources in CSV. 2: Logic Tier containing all libraries in pithing and R used for processing and manipulation. 3: Presentation Layer where the modeling algorithms, prediction and forecasts are presented.

5. Implementation and Solution Development

This section describes the solution implementation based on the methodology described in section 3 above. Also visualization and schematics are presented where necessary to provide clarity. First, the solution architecture is presented in Figure 3 below:

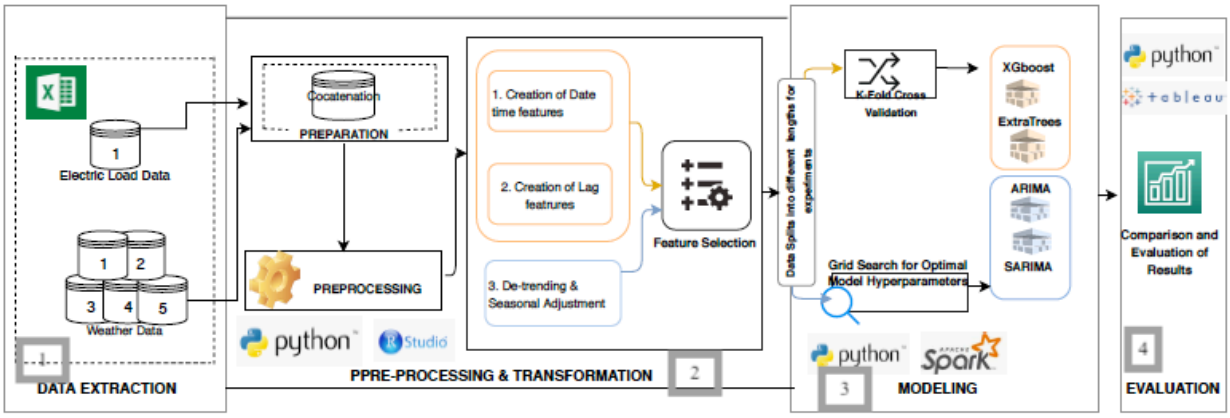

Figure 3: Implementation of the day-ahead electric load forecast system

1: Data Extraction (Load and weather data). 2: Preprocessing and Transformation consisting creation of date time features, creation of lag features, de-trending and seasonal adjustment, feature selection. 3: Modeling showing cross validation, grid search and algorithms used. 4: Evaluation and Results.

5.1 Data Exploration, Preprocessing and Transformation

A total of six datasets were extracted for experimental use in csv format—five of them being hourly weather readings for various cities in North America with date-time features, while the last dataset records the hourly power consumption (in megawatts) for various cities in North America.

Extraction and Cleaning using R Studio: All six datasets were imported into R studio for cleaning and initial processing. For load consumption, the raw data file consisted of 2 variables (Datetime and Power_MW), and 116, 189 observations after missing values were dropped. The raw weather files namely temperature, humidity, pressure, wind speed, and wind direction each contained 37 columns that included datetime and the corresponding weather measurements for 36 cities. Our interest is for North Carolina so all other cities were dropped. Also all weather datasets had no missing values except for the pressure dataset (which had only 3 missing values), leaving us with: 2 variables and 45,250 observations for temperature; 2 variables and 45,250 observations for pressure; 2 variables and 45,252 observations for wind direction; 2 variables and 45, 253 observations for humidity; 2 variables and 45, 251 observations for wind speed. The datasets were then exported to the Jupyter notebook for python where exploratory data analysis and further feature engineering was implemented.

Exploratory Analysis: In python, necessary data analysis libraries were installed namely numpy, pandas, seaborn, matplotlib, xgboost, sklearn, etc. Using the pandas library *concat function*, all six separate datasets were concatenated on the basis of the datetime column resulting in one single dataset of 7 variables (datetime, Power_MW, temperature, wind speed, pressure, wind direction, humidity), upon which our analysis were carried out.

Stationarity of time series data is a critical assumption for statistical modeling methods such as ARIMA. Therefore, it is important to make preliminary checks for the presence of seasonal or trend effects in the time series datasets early enough in the exploration stage. This is such that necessary seasonal or trend adjustments techniques can be applied in the feature-engineering phase. [31] recommended three approaches to check for stationarity namely: inspection of plots, summary statistics, and a statistical test (The Augmented Dickey-Fuller Test).

- **Plots:** A visual inspection of the line plot of the time series dataset presented in Figure 4 shows cycles giving rise to a possible presence of seasonal or trend elements. Also autocorrelation plots can provide an indication of the presence or not of

seasonality in time series data. In Figure 5, the autocorrelation plot is shown on the y-axis and lagging features on the x-axis. The graph captures the self-correlation between an observation and its equivalent in similar or opposite seasons. Sinusoidal wavelike distributions such as these are a possible indication of seasonality.

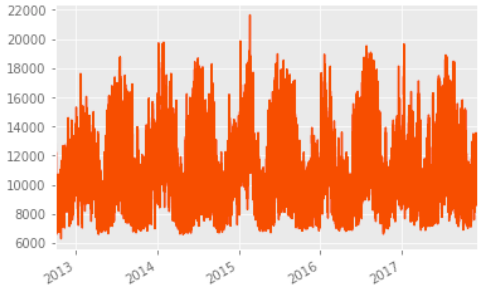

Figure 4: Line plot for the time series

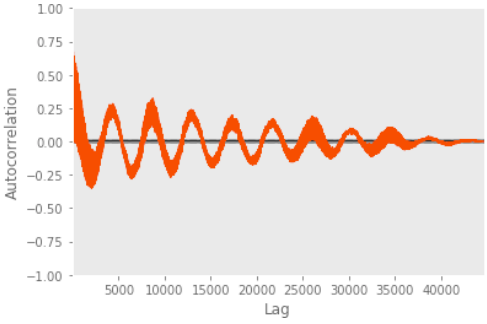

Figure 5: Autocorrelation plot

- **Summary Statistics:** A review of summary statistics is another quick way to inspect if a time series data is non-stationary. To achieve this, the time series was split into two partitions X1 and X2, inspecting the summary statistics of each partition to compare their mean and variance. Figure 6 shows the summary statistics presented for both partitions, with the mean and variance not showing significant spread.

```
mean1=10984.657142 mean2=11008.380330
variance1=5492529.712206 variance2=6040740.843794
```

Figure 6: Stationarity by summary statistics

```
ADF Statictics: -12.411948
p-value: 0.000000
Critical values:
        1%: -3.430
        5%: -2.862
        10%: -2.567
```

Figure 7: ADF test statistics

- **The Augmented Dickey-Fuller Test**: Is one of the most widely used unit root test to inspect how strongly a trend defines a time series. It tests the following hypothesis:

  (H_0) = the time series data is non-stationary because of the presence of a unit root
  (H_1) = the time series is stationary due to the absence of a unit root.

The test is interpreted using the p-value. If P > 0.05, we fail to reject the null hypothesis (i.e the timer series non-stationary). Whereas if P < 0.05 we reject the null hypothesis (i.e the time series is stationary). Figure 7 above shows the result of the test. Our P= 0.000, <0.05 further confirming our time series is stationary. The *adfuller ()* function in python was used to carry out this task.

Transformation: A number of feature engineering techniques were applied to further validate necessary assumptions as well as prepare the datasets for modeling. [31] recommended that a time series dataset must be further transformed so as to effectively model it as a supervised learning problem. To achieve this task, new features were invented from our time series dataset. The tasks carried out here are summarised below:

- **Creation of Date-Time Features:** The Pandas library in python was used to extract further features from the time-stamp of our data, adding further columns one at a time such as day of week, quarter, month, day of year, hour of day, leap year or not, etc.

- **Creation of Lag Features:** The classical way to transform time series for effective use in supervised learning problem is to use lag features. This method involves predicting the next time value (t +1) from the current time (t). This creates a supervised learning problem with shifted values. For this research, the Pandas library's shift () function was used to create seven lags from our dataset.

- **De-trending & Seasonal Adjustment using Differencing:** Differencing was used to carry out de-trending and seasonal adjustment on the datasets. Below Figure 8 shows the original data, Figure 9 the line plot for the de-trended data, while Figure 10 is the line for the seasonally adjusted data.

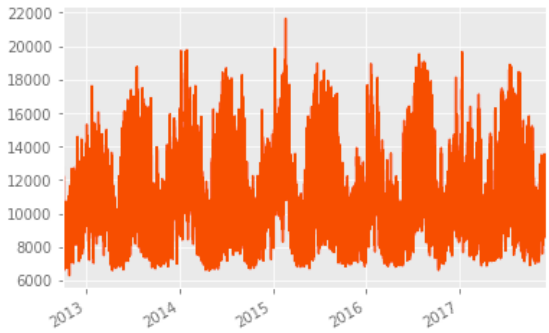

Figure 8: Load Time Series

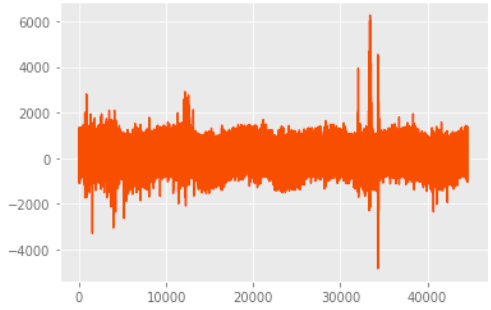

Figure 9: De-trended Load

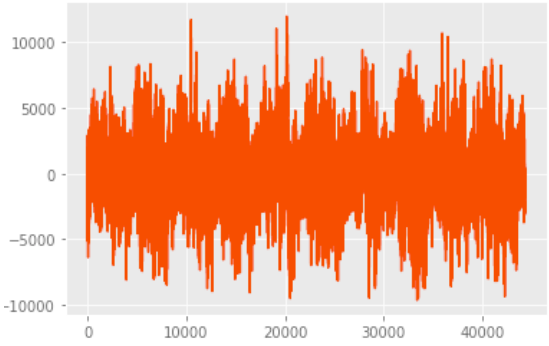

Figure 10: Seasonally Adjusted Load

Feature Selection: A feature importance plot of F-score using Xgboost regression modeling was used to identify the most important features. Figure 11 below presents the ranking by F-score of the 18 features selected out of the total 54 features going forward.

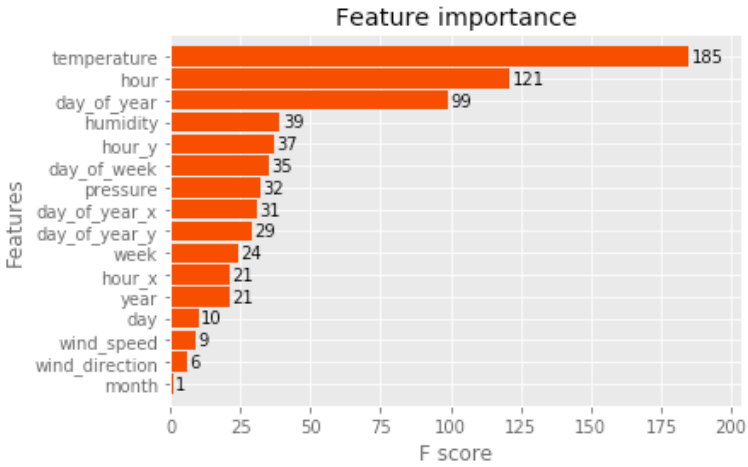

Figure 11: Feature Selection

Summarily, the transformation process addressed one half of the model fitting approach by achieving three things namely: stationarity of the time series data—a critical assumption for applying autoregressive modeling techniques, creation of lagging features—necessary for adapting the time series to regression-based supervised learning modeling techniques, and lastly selecting the best features to be used for load forecasts.

5.2 Application of Data Mining Algorithms

Upon successful completion of the transformation steps, a total of four modeling techniques namely ARIMA, SARIMA, Xgboost, Extra Trees Regressor were used to forecast day-ahead load using different lengths of data (i.e 2 weeks, 4 weeks, 6 months, 1 year and 6 years). Our data was split into 70% for training and 30% for testing. K-fold cross validation was used for validation as extensively discussed in section 3 above.

6. Results and Evaluation

Results obtained from modeling with different algorithms described in section 5 above on datasets presented in section 3.1 are presented in this section. In summary, the experiments performed were aimed at three focal points: first, to ascertain which modeling technique is best suited to predicting day-ahead electric load consumption. Secondly to find the data threshold (length of data) required for achieving best performance for each of the models. Lastly, to deduce the impact of distributed systems on processing time for large-scale data.

For performance measurements, error metrics such as MAPE, RMSE and MAE were used. The accuracy metrics are described below:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(Y_{obs,i} - Y_{model,i})^2}{n}} \quad (4)$$

$$MAPE = \frac{100}{n} \left| \frac{\sum_{i=1}^{n} Y_{obs,i} - Y_{model,i}}{Y_{obs,i}} \right| \quad (5)$$

$$MAE = \left| \frac{\sum_{i=1}^{n} Y_{obs,i} - Y_{model,i}}{Y_{obs,i}} \right| \quad (6)$$

Where $Y_{obs,i}$ is the actual value, and $Y_{model,i}$ is the forecast value.

It is worthy of note that both **RMSE** and **MAE** error values are the units as the dependent variable. As such, it is necessary to normalize the error values to get the actual errors as presented in equation 7 below:

$$\text{Error}_{\text{actual}} = \frac{Error}{\sigma} \quad (7)$$

Also, these two metrics have been widely used to compare the performances of different time series forecasting approaches [28, 29]. The closer to zero the error values are, the better the model. RMSE is particularly useful when avoiding large forecasting errors as it assigns a big penalty to enormous errors. Results are presented in tables 2, 3 and 4 below:

Data Length	6 Years		1 year		6 Months		4 Weeks	
Model	XGBOOST	ExtraTrees	XGBOOST	ExtraTrees	XGBOOST	ExtraTrees	XGBOOST	ExtraTrees
RMSE	0.492	0.465	1.099	1.524	0.825	0.920	0.69	0.849
MAPE	7.66	7.037	17.471	22.99	11.605	12.387	5.899	7.238
MAE	0.366	0.334	0.935	1.251	0.588	0.637	0.529	0.649
C.V/Accuracy	0.73	0.71	0.60	0.58	0.61	0.53	0.49	0.46
Confidence Interval	(+- 0.21)	(+- 0.19)	(+- 0.18)	(+- 0.18)	(+- 0.11)	(+- 0.13)	(+- 0.27)	(+- 0.20)
Processing Time	3.64 seconds	1.87 seconds	786 ms	364ms	476ms	211ms	97.3ms	40.8ms

Table 2: Results for regression-based machine learning models

Data Length	6 Years	1 year	6 Months	4 Weeks
SARIMA	(1,1,1)(0,0,1)12	(1,0,1)(1,1,1)12	(0,1,1)(1,1,2)12	(1,1,1)(1,1,1)12
RMSE	0.574	0.800	0.525	0.129
MAPE	11.278	29.472	5.866	1.080
MAE	0.4	0.448	0.408	0.094
Processing Time	544seonds	92 seconds	21.1 seconds	491seconds

Table 3: Results for SARIMA

Data Length	1 Week	4 Weeks
ARIMA	(6,1,0)	(6,0,2)
RMSE	0.146	0.161
MAPE	14.979	13.257
MAE	0.109	0.125
Processing Time	152seconds	765 seconds

Table 4: Results for ARIMA

The next section of this documentation presents the experiments and outputs from **ARIMA**, **SARIMA, Xgboost, and Extra Trees Regressor.** Day-ahead forecast experiments were carried out for four scenarios of data length namely: 4 weeks, 6months, 1 year and 6 years. (ARIMA was modeled for 1 and 4 weeks length data only)

6.1 Experiment 1 (Day-Ahead load forecast using XGBoost Regressor):

The first experiment is to evaluate the performance of Xgboost on forecasting day-ahead load given different scenarios of data lengths. The error metrics RMSE, MAPE and MAE are used to ascertain the goodness of fit of Xgboost in modeling these scenarios as presented in results table 3 above. Also the processing time for prediction is recorded. Figure 12 shows the outputs for Xgboost on 6 years, 1 year, 6 months and 4 weeks length of data respectively.

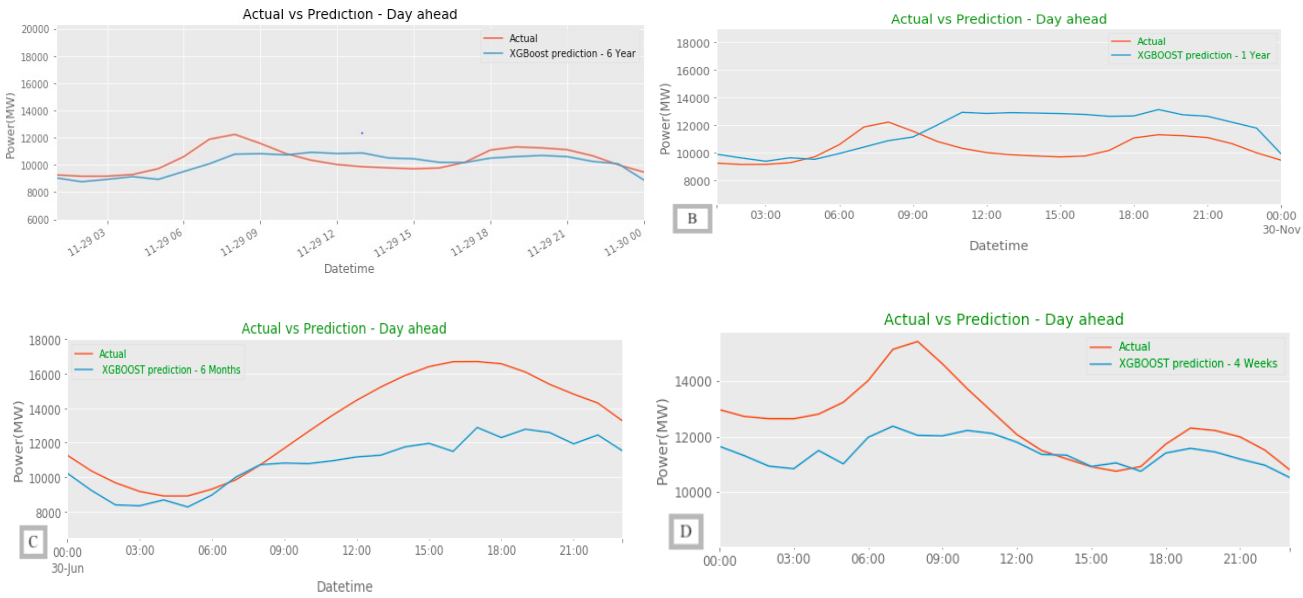

Figure 12: Experimental outputs for XGBoost Regression

It follows therefore that for day-ahead forecast of electric load, Xgboost Regressor shows the best performance—as seen in figure 12a above, and RMSE of 0.492 in modeling longer lengths of data (6 years).

6.2 Experiment 2 (Day-Ahead load forecast using Extra Trees Regressor)

In the second experiment, similar procedure as section 6.1 above was carried out on the same time series data using Extra Trees Regressor model. Day-ahead forecasts are presented in figure 13 for 6 years, 1 year, 6 months, and 4 weeks length data respectively.

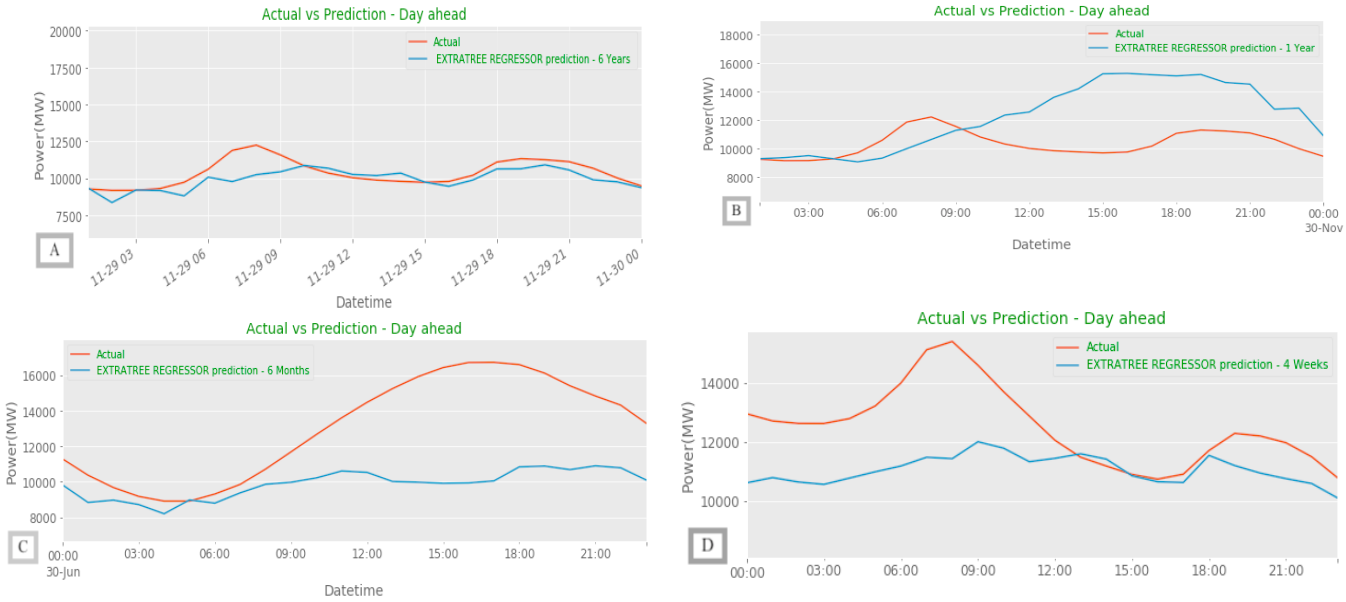

Figure 13: Experimental outputs for Extra Trees Regressor Model

Comparing the outputs in figure 13 above, the Extra Tree Regressor model shows the best performance in forecasting dah-ahead load with longer data lengths (6 years time series), recording the lowers RMSE of 0.465. This is similar to the observed trends using Xgboost.

6.3 Experiment 3 (Day-Ahead load forecast using SARIMA)

In the third experiment, SARIMA was used to build day-ahead forecast models for the different lengths of data. However, the procedure here was a little more complicated than the regression based experiments described in 6.1 and 6.2 above. To begin each experiment, a seasonal decompose procedure is used to split the time series data into its trend, seasonality and residual elements as shown in figure 14 below for the 6 years data length.

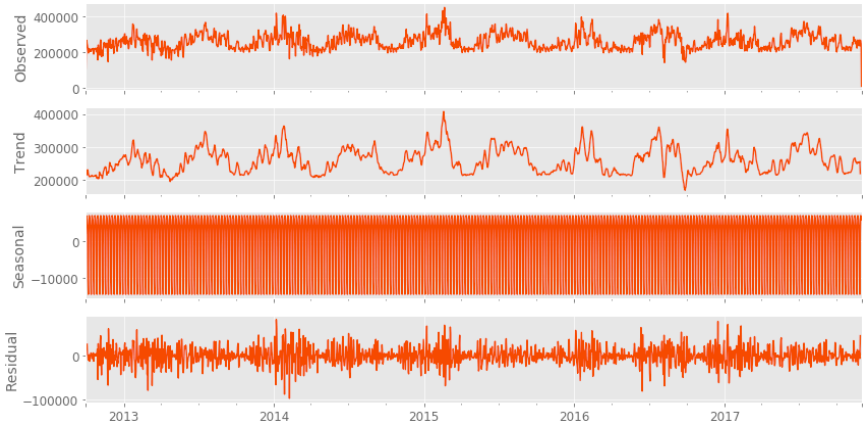

Figure 14: Time series decomposed plot for six years data

The decompose graphs for the rest of the data lengths are presented in the configuration manual attached to this research document.

Next, for each data length scenario, a grid search procedure was used to find the optimal [(p, d, q) x (P, D, Q) m] hyper parameters to be used. This also doubles as the validation technique for autoregressive modeling methods, as the combination of parameters with the lowest **AIC score** is selected for building the forecast model. [31]. As in the previous experiments, forecast results are presented in Figure 15 for the respective data lengths.

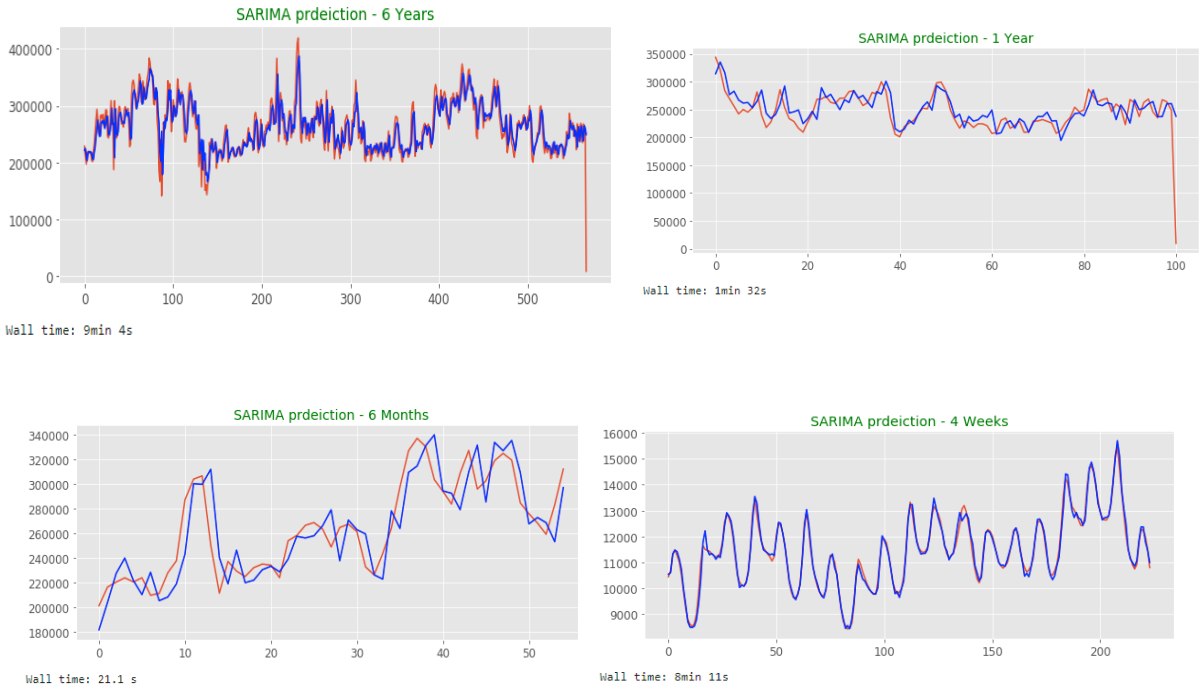

Figure 15: Experimental outputs for SARIMA Model

The outputs in figure 15 above show that the SARIMA model improves with shorter time series data lengths. The model shows overall best performance for the 4 week length time

series data with an RMSE of 0.126 outperforming the regression models.

6.4 Experiment 4 (Day-Ahead load forecast using ARIMA)

Similar experiments as described in section 6.3 above were conducted on the datasets using ARIMA. However, ARIMA struggled with building forecast models for data lengths beyond 4 weeks. As such for this experiment, ARIMA was used to build day-ahead forecast models using 2 weeks length and 4 weeks lengths data. Outputs are presented in figures 16a and 16b below:

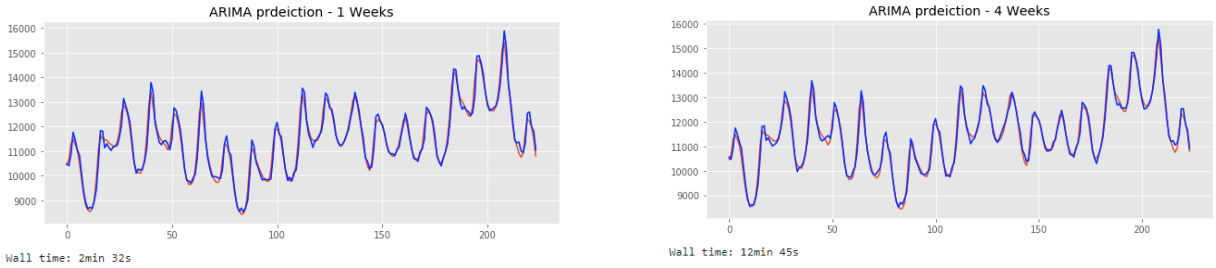

Figure 16: Experimental outputs for ARIMA Model

7.0 Discussion

This section presents detailed discussion on the findings from the experiments presented in section 6 above. Specifically, findings from this research are explained based on output from the experiments, as well as benchmarked against findings from other literature.

Observations from experiments 1 and 2 as described in sections 6.1and 6.2 above, where regression techniques like Xgboost and Extratrees Regressor were used to model day-ahead load forecasts for different lengths of historical data show that regression techniques generally performed well in modeling longer data lengths beyond 1 year (say from 2 to 6 years). Results show that both the Xgboost and Extra Trees Regressor outperformed autoregressive models like SARIMA for lengthier historical data of 6 years recording RMSE values of 0.492 and 0.465 respectively, compared to an RMSE 0.574 for SARIMA (1,1,1)(0,0,1) 12. For the two regression based models, Extratrees Regressor particularly showed the best performance for lengthier historical data, outperforming Xgboost in the respective RMSE values for 6 years historical data. Also in terms of processing time, Extra trees recorded 9.8seconds compared to 3.64 seconds for Xgboost. Consequently, the regressor models show effective adaptation in dealing with time-series fluctuation given implementation of necessary feature engineering tasks such as lag features, and are well

suited to managing extreme values. Also the models performance is highly dependent on explanatory variables. However, performance of both regression models tapered off with shorter historical data lengths as RMSE values increased to 0.690 and 0.849 for Xgboost and Extra Trees regressor for the 4-week data historical data length.

Results for autoregressive modeling techniques presented in 6.3 and 6.4 show the results for SARIMA and ARIMA models respectively. As observed, these models generally outperformed the regression-based models for day-ahead load forecasts with shorter lengths of historical data. Particularly SARIMA (1,1,1)(1,1,1) 12 showed the best overall performance for modeling 4 week-length historical data with an RMSE of 0.129 (the lowest for all experiments). That said, the observed trend from our results show that these models struggle with increased length of historical data as ARIMA was only able to successfully model not more than 4 weeks data length, while SARIMA recorded the highest RMSE 0.800 for 1 year data length the entire research. These results make sense as classical statistical models are frequently applied to model short-term load forecasts using shorter lengths of historical data. Also the models were able to effectively model the trend and seasonality elements.

Specifically, to address our research questions, Table 5 below captures the summary of findings from the experiments carried out. Stakeholders are able to reference this to decide what modeling approaches best optimize electric load forecasts giving various scenarios.

Approach	Model	Pros	Cons
Classical Statistical Methods	ARIMA SARIMA	(i) Delivers great accuracy for shorter time series history and data length (4 to 6 months) (ii) Requires fewer parameters to estimate and results are easy to interpret. (iii) ARIMA delivers even better for modeling very short data length due to the absence of seasonal elements (1 to 4 weeks)	(i) Accuracy lowers with increased historical data length. (ii) More complicated to implement and interpret the results, as a rigorous process is required to select the best hyperparameters.
Regression Based Machine Learning	Xgboost Extra Trees	(i) Delivers great accuracy for longer time series history and data length (2 years and above). (ii) It's easy to implement, interpret and understand.	(i) Largely dependent on explanatory variables (ii) Modeling requires reframing the time series into a supervised learning problem by applying feature engineering techniques like lagged features

Table 5: Summary of findings from research

8. Conclusion and Future Work

In this research work, different approaches to electric load forecasting were investigated with the aim of ascertaining the goodness of fit of each model on different historical data lengths that include 6 years, 1 year, 6 months, and 4 weeks. Two broad categories of modeling approaches were used namely classical statistical approach—ARIMA and SARIMA, and regression based machine-learning approach—Xgboost and Extra Trees Regressor. The comparative analysis was achieved using a 10-year electric load data from PJM and weather data. Upon successful completion of all experiments in data pre- processing, feature engineering, and modeling, the results are presented. The key finding is that machine learning regression models delivered the best performance for lengthier historical time series—specifically Extra trees regressor with an RMSE of 0.465 for 6 years historical data length. This is in addition to the performance in terms of processing time, which surpassed all other models at 1.8 seconds for the 6 years data length. However, the autoregressive models delivered better results for day-ahead load forecast on shorter length of time series data. Especially SARIMA delivered the best performance overall achieving an RMSE of 0.129 for 4 weeks data length. Therefore a safe historical data threshold for applying regression based models for effective performance is 2 years or above, while the data should between 4 weeks and 6 months to get efficient results for SARIMA. For ARIMA, a data length of 1 to 4 weeks delivered good results.

For future work, the approaches deployed in this research should be applied to evaluate performance of neural network based algorithms on different lengths of data. Particularly as deep learning shows good performance with time series analysis.

Acknowledgement

First, I'd like to show appreciation to the Government of Ireland for taking necessary initiatives for capacity building in STEM allowing for international student to further develop and horn their skills in pursuing a career in tech. Next, I'd like to thank the National College of Ireland for their approach to dispensing knowledge especially for delivering curriculums that are industry-specific and relevant. I also will like to show gratitude to all my lecturers, and course mates who have assisted in the knowledge process over the past year, and finally my supervisor Dr. Vladimir Milosavljevic for the support and resources through the project.

Thank you!

References

[1] Gooijer, J.G.D.; Hyndman, R.J. 25 years of time series forecasting. Int. J. Forecast. 2006, 22, 443–473.

[2] Abdel-Aal, R.E.; Al-Garni, A.Z. Forecasting monthly electric energy consumption in eastern Saudi Arabia using univariate time-series analysis. Energy 1997, 22, 1059–1069.

[3] Shilpa, G.N.; Sheshadri, G.S. Short-Term Load Forecasting Using ARIMA Model For Karnataka State Electrical Load. Int. J. Eng. Res. Dev. 2017, 13, 75–79.

[4] Chujai, P.; Kerdprasop, N.; Kerdprasop, K. Time Series Analysis of Household Electric Consumption with ARIMA and ARMA Models. In Proceedings of the International Multi conference of Engineers and Computer Scientists, Hong Kong, China, 13–15 March 2013; Volume 1.

[5] Rallapallia, S.R.; Ghosh, S. Forecasting monthly peak demand of electricity in India—A critique. Energy Policy 2012, 45, 516–520.

[6] Newsham, G.R.; Birt, B.J. Building-level Occupancy Data to Improve ARIMA-based Electricity Use Forecasts. In Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building (BuildSys '10), Zurich, Switzerland, 2 November 2010; ACM: New York, NY, USA, 2010; pp. 13–18.

[7] Nowotarski, J.; Liu, B.;Weron, R.; Hong, T. Improving short term load forecast accuracy via combining sister forecasts. Energies 2016, 98, 40–49.

[8] Rahman, H.; Selvarasan, I.; Begum, J. Short-Term Forecasting of Total Energy Consumption for India-A Black Box based Approach. Energies 2018, 11, 3442–3463.

[9] Schrock, D.W.; Claridge, D.E. Predicting Energy Usage in a Supermarket. In Proceedings of the 6th Symposium on Improving Building Systems in Hot and Humid Climates, Dallas, TX, USA, 3–4 October 1989.

[10] Samarasinghe, M.; Al-Hawani, W. Short-Term Forecasting of Electricity Consumption using Gaussian Processes. Master's Thesis, University of Agder, Kristiansand, Norway, 2012.

[11] Ali, D.; Yohanna, M.; Puwu, M.I.; Garkida, B.M. Long-term load forecast modeling using a fuzzy logic approach. Pac. Sci. Rev. A 2016, 18, 123–127.

[12] Park, D.C.; El-Sharkawi, M.A.; Marks, R.J.; Atlas, L.E.; Damborg, M.J. Electric load forecasting using an artificial neural network. IEEE Trans. Power Syst. 1991, 6, 442–449.

[13] Khotanzad, A.; Hwang, R.-C.; Abaye, A.; Maratukulam, D. An adaptive modular artificial neural network hourly load forecaster and its implementation at electric utilities. IEEE Trans. Power Syst. 1995, 10, 1716–1722.

[14 Javed, F.; Arshad, N.; Wallin, F.; Vassileva, I.; Dahlquist, E. Forecasting for demand response in smart grids: An analysis on use of anthropologic and structural data and short term multiple loads forecasting. Appl. Energy 2012, 96, 150–160.

[15] Lamedica, R.; Prudenzi, A.; Sforna, M.; Caciotta, M.; Cencellli, V.O. A neural network based technique for short-term forecasting of anomalous load periods. IEEE Trans. Power Syst. 1996, 11,

1749–1756.

[16] Shi, H.; Xu, M.; Li, R. Deep Learning for Household Load Forecasting—A Novel Pooling Deep RNN. IEEE Trans. Smart Grid 2017.

[17] Ertugrul, Ö.F. Forecasting electricity load by a novel recurrent extreme learning Machines approach. Int. J. Electr. Power Energy Syst. 2016, 78, 429–435.

[18] Al-Musaylh, M.S.; Deo, R.C.; Adamowski, J.F.; Li, Y. Short-term electricity demand forecasting with MARS,SVR and ARIMA models using aggregated demand data in Queensland, Australia. Adv. Eng. Inf. 2018, 35, 1–16.

[19] El-Telbany, M.; El-Karmi, F. Short-term forecasting of Jordanian electricity demand using particle swarm optimization. Electr. Power Syst. Res. 2008, 78, 425–433

[20] Liu, N.; Tang, Q.; Zhang, J.; Fan, W.; Liu, J. A hybrid forecasting model with parameter optimization for short-term load forecasting of micro-grids. Appl. Energy 2014, 129, 336–345.

[21] Hong, W.-C. Chaotic particle swarm optimization algorithm in a support vector regression electric load forecasting model. Energy Convers. Manag. 2009, 50, 105–117

[22] Guan, C.; Luh, P.B.; Michel, L.D.; Wang, Y.; Friedland, P.B. Very short-term load forecasting: Wavelet neural networks with data pre-filtering. IEEE Trans. Power Syst. 2013, 28, 30–41.

[23] A. Mellit, M. Benghanem, and S.A. Kalogirou, "An adaptive wavelet-network model for forecasting daily total solar-radiation", Applied Energy, Vol. 83, Issue 7, July 2006, pp. 705-722, Elsevier

[24]Box, G.; Jenkins, G. Time Series Analysis: Forecasting and Control; John Wiley and Sons: New York, NY, USA, 2008.

[25] Nie, Hongzhan, Guohui Liu, Xiaoman Liu, and Yong Wang.(2012) "Hybrid of ARIMA and SVMs for Short-Term Load Forecasting" *Energy Procedia*, **(16)**. Doi: 10.1016/j.egypro.2012.01.229

[26] Paul S.P. Cowpertwait, Andrew V. Metcalfe. "Introductory Time Series with R!", Chapter 7 Non stationary Models.(2009)

[27] Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDDInternational Conference on Knowledge Discovery and Data Mining (KDD '16), San Francisco, CA, USA,13–17August 2016; ACM: New York, NY, USA, 2016; pp. 785–794.

[28] Talavera-Llames, R.; Pérez-Chacón, R.; Troncoso, A.; Martínez-Álvarez, F. MV-kWNN: A novel multivariate and multi-output weighted nearest neighbours algorithm for big data time series forecasting. Neurocomputing 2019, 353, 56–73.

[29] Galicia, A.; Talavera-Llames, R.; Troncoso, A.; Koprinska, I.; Martínez-Álvarez, F. Multi-step forecasting for big data time series based on ensemble learning. Knowl.-Based Syst. 2019, 163, 830–841.

[31] Brownlee J. Machine learning mastery: Introduction to Time Series Forecasting with Python. How to prepare data and develop models to predict the future.

[32] Max A Little, Gael Varoquaux, Sohrab Saeb, Luca Lonini, Arun Jayaraman, David C Mohr, Konrad P Kording, Using and understanding cross-validation strategies. Perspectives on Saeb et al., *GigaScience*, Volume 6, Issue 5, May 2017, gix020,