International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 05 | May - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Machine learning applied to electric load forecast: A study comparing results obtained from various regression methods

Himani Kulshreshtha[1], Prof. Vishal V.Mehtre[2]

[1]Department of Electrical Engineering, Bharati Vidyapeeth Deemed University, COE, Pune
[2]Department of Electrical Engineering, Bharati Vidyapeeth Deemed University, COE,Pune

---

Abstract - Short-term electricity load forecasting is pivotal in efficient energy management and resource allocation within power systems. In this research paper, we present a comparative analysis of regression methods for short-term electricity load forecasting, implemented in Python using machine learning techniques. The study explores the performance of various regression models including linear regression, support vector regression, decision tree regression, and random forest regression. Historical load data from a real-world electricity grid is utilized for training and evaluating the forecasting models. The evaluation metrics considered include Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) to assess the accuracy and robustness of the models. Through extensive experimentation and analysis, insights into the strengths and weaknesses of each regression method in the context of short-term electricity load forecasting are provided. The findings of this study contribute to the advancement of predictive modeling techniques in the domain of energy forecasting and facilitate informed decision-making for energy stakeholders.

Key Words: Short-term electricity load forecasting, Machine Learning, Python, Regression methods

I. INTRODUCTION

The accurate forecasting of electricity load is crucial for efficient resource allocation, infrastructure planning, and energy management in modern power systems. Short-term load forecasting, in particular, plays a pivotal role in ensuring grid stability and reliability. In this research paper, we delve into the comparative analysis of regression methods for short-term electricity load forecasting, leveraging the power of machine learning techniques implemented in Python. The focus of this study is on evaluating the efficacy of various regression algorithms in predicting electricity load demand within Panama City. The dataset utilized for this analysis is sourced from Kaggle, providing comprehensive historical data on electricity consumption patterns, weather conditions, and other relevant variables. Panama City's dataset offers a rich source of real-world data, facilitating a robust evaluation of forecasting models in a dynamic urban environment.

Through this research, we aim to assess the performance of regression methods such as linear regression, support vector regression, decision tree regression, and random forest regression in predicting short-term electricity load demand. By comparing these methods' accuracy, robustness, and computational efficiency, we endeavor to identify the most suitable approach for electricity load forecasting in Panama City's energy landscape. This study contributes to advancing the domain of energy forecasting and provides valuable insights for policymakers, utility companies, and researchers striving for optimized energy management strategies.

The remainder of this paper is structured as follows: Section II discusses related works focusing on load prediction and other methods being used for the same. In Section III, we define the proposed approach and used methods. Section IV introduces the simulations and numerical results. Then in Section V, we discuss the limitations and future work. Section VI concludes the paper.

II. RELATED WORK

Many recent research works used deep neural networks, and particularly Long-short term memory (LSTM) to tackle the short-term load forecasting challenge. Benchmarks have proved LSTM's potential compared to other methods [12,13], yet the results do not match the level of desired exactitude in terms of Root Mean Square Error (RMSE) and Mean Average Percentage Error (MAPE).

To show the importance of meteorological parameters the author [3] introduces a Machine Learning Based Cost Effective Electricity Load Forecasting Model utilizing Correlated Meteorological Parameters. Focusing on accurate load forecasting's importance, it proposes the lcELFM algorithm, aiming to minimize the unit price by considering meteorological factors. Employing MLR, kNN, SVR, Random Forest, and AdaBoost, it analyzes correlations between load and meteorological data, significantly improving forecast accuracy by integrating weather variables. Correlation analysis confirmed a strong linear relationship between meteorological factors and electric load demand, enhancing forecasting accuracy when integrated. Meteorological parameters significantly influence electricity consumption, impacting load demand. Variables like temperature, humidity, and wind speed affect consumption, with inclusion in forecasting models improving accuracy and providing insights into demand fluctuations. Another author [6,7] investigates factors affecting electric load forecasting, categorizing them into meteorological, temporal, economic, random, and customer factors. These include meteorological factors such as rain, snowfall, wind speed, temperature, and humidity. Temporal and calendar factors, such as working days, moving holidays, and time of day, also play a significant role. Additionally, economic factors based on regional or national economic activities influence load forecasting. Random factors, such as sudden variations in load consumption due to large-scale industry and agricultural load, are also considered. Other

factors, such as the load variation curve in different parts of the country, also contribute to the complexity of load forecasting. Temperature emerges as a significant climatic factor influencing load forecasting, leading to increased energy consumption. The research underscores the importance of accurate forecasting for effective energy management and system planning.

Some authors [2], explore a transformer-based model for electrical load forecasting, crucial in energy management amidst rising consumption and emission reduction goals. It adapts the transformer architecture, commonly used in NLP, enhancing it with contextual modules and N-space transformation. Results show superior performance over state-of-the-art models, particularly for longer forecasting horizons and diverse data streams.

The hybrid load demand forecast model [1], incorporating LSTM and RNN architectures, shows superior accuracy and generalization. Particularly, the combination of LSTM and RNN outperforms conventional ANN designs, with the RNN model achieving 1.01% MAPE for one-day ahead forecasts. Hybrid models maintain reasonable accuracy across all seasons, demonstrating their effectiveness under varying load demands year-round. Another such hybrid approach is suggested by the author [4] which explores short-term load forecasting using machine learning techniques, introducing PLCNet, a novel parallel deep LSTM-CNN approach. Evaluation of models like ARIMA, Exponential Smoothing, Linear Regression, and PLCNet with real-world data from Malaysia and Germany emphasizes accurate predictions' critical role in energy management. Results showcase PLCNet's potential in enhancing forecasting accuracy. Another study [5], presents a hybrid model, combining CART and DBN, to enhance daily load forecasting accuracy amidst external factors like weather and time. Compared to standalone models, the hybrid model demonstrates superior performance, emphasizing the significance of understanding load patterns and employing pattern classification for improved forecasting.

Further researchers [8] explore short-term electrical load forecasting using time series analysis, focusing on ARMA, ARIMA, and ARIMAX models. The study emphasizes their significance in power system operation, particularly in developing countries like India. The overall forecasting error is reduced to 3.6% using ARIMAX. The forecasted error of ARMA is 17.7% & ARIMA is 4%. Hence, a comparison of time series models shows that ARIMA performance is better than ARMA, and ARIMAX forecasting results are better than ARIMA. The experimentation concludes by highlighting the significance of the proposed methodology in accurately forecasting electrical loads and its potential to improve forecasting accuracy, especially when considering exogenous variables like weekdays, weekends, and public holidays, in addition to load and time, as inputs to the ARIMAX model. It underscores the potential for further improving load forecasting accuracy by considering weather variables and the dynamic nature of load profiles with temporal, seasonal, and annual variations.

Another author [9] says, however, that Deep neural networks excel in short-term load forecasting (STLF) at the household level due to their capacity to capture intricate patterns, outperforming methods like ARIMA and SVR. Even though he mentions, solely relying on deep learning may lead to overfitting, hindering generalization to new data as it learns noise details from training data. His research addresses load forecasting challenges in the smart grid and proposes edge computing and federated learning as solutions. It emphasizes privacy concerns and accuracy requirements, evaluating their effectiveness in household load forecasting. The study highlights potential gains in accuracy, network load reduction, and data privacy preservation, offering valuable insights for smart grid optimization. The study evaluates the use of edge computing and federated learning for household load forecasting, achieving promising results using data from 200 houses in Texas, USA.

In another load forecasting experiment [10], the Mean Absolute Percentage Error (MAPE) was used to compare the performance of different models, including ARIMA, SVM, LSTM, and CNN-LSTM. The results showed that the deep learning models, particularly LSTM and CNN-LSTM, outperformed the traditional linear model (ARIMA) and the SVM model. Specifically, the MAPE values for one day ahead and six months ahead predictions were as follows:

Regression Method	Results
ARIMA	0.2012 and 0.1989
SVM	0.1853 and 0.1869
LSTM	0.1567 and 0.1601
CNN-LSTM	0.0985 and 0.0856

Table 1: Shows the comparative results of various methods.

The comparison revealed that the deep learning models, especially the ensemble model represented by CNN-LSTM, achieved significantly lower MAPE values, indicating higher accuracy and better performance in load forecasting compared to the traditional linear model and the SVM model. Specifically, the MAPE value of the deep learning models was reduced by 22.1% and 19.5% for one-day ahead predictions, and by 51.0% and 57.0% for six-month ahead predictions, demonstrating a substantial performance improvement. These findings highlight the superior predictive capabilities of deep learning models, particularly the CNN-LSTM ensemble model, in load forecasting experiments.

III. SYSTEM MODEL

A. Outliers

An outlier refers to a data point that notably differs from the rest of the dataset, often termed a normal object. Recognizing outliers holds significance in statistics and data analysis since they can substantially influence statistical outcomes. Outlier detection, also known as outlier mining, aims to identify such data points. Outliers can distort the mean and impact measures of central tendency, along with affecting the outcomes of statistical significance tests. These outliers are caused due to measurement errors, sampling errors, natural variability, data entry errors, experimental errors, sampling from, multiple populations or intentional outliers, etc.

We first visualize the data with box plots by having days of the week on the x-axis and energy consumption(MW) on the y-axis. The boxplot succinctly presents data summaries through a straightforward depiction of a box and whiskers. It efficiently represents sample data by showcasing quartiles at the 25th, 50th, and 75th percentiles. Merely glancing at the boxplot

provides insights into the dataset's quartiles, median, and identification of outliers (refer to Fig. 1 and Fig. 2).

We use visual representation of box plots for the detection of outliers and the removal of outliers we use the IQR method which is used to measure variability by dividing a data set into quartiles. The data is sorted in ascending order and split into 4 equal parts. Q1, Q2, and Q3 called the first, second, and third quartiles are the values that separate the 4 equal parts.

- Q1 represents the 25th percentile of the data.
- Q2 represents the 50th percentile of the data.
- Q3 represents the 75th percentile of the data.

If a dataset has 2n or 2n+1 data points, then

- Q2 = median of the dataset.
- Q1 = median of n smallest data points.
- Q3 = median of n highest data points.
- IQR = Q3 — Q1

To identify outliers using the IQR method, we establish two boundaries:

- Lower Bound: Q1–1.5 * IQR
- Upper Bound: Q3 + 1.5 * IQR

These boundaries help us determine which data points might be outliers. Any data point that falls below the lower bound (Q1–1.5 * IQR) is considered an outlier. These values are significantly lower than the majority of the dataset and are potential candidates for removal or further investigation.

Conversely, any data point that exceeds the upper bound (Q3 + 1.5 * IQR) is also considered an outlier. These values are much higher than most of the dataset and may warrant special attention.

B. Machine Learning Algorithms

i. Simple Linear Regression

Simple Linear Regression is a type of Regression algorithm that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The Simple Linear Regression model can be represented using the below equation:

$y= a0+a1x+ \varepsilon$

Where,

- a0= It is the intercept of the Regression line (can be obtained by putting x=0)
- a1= It is the slope of the regression line, which tells whether the line is increasing or decreasing.
- ε = The error term. (For a good model it will be negligible)

ii. Multiple Linear Regression

In the previous topic, we learned about Simple Linear Regression, where a single Independent/Predictor(X) variable is used to model the response variable (Y). However, there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.

We can define it as:

"Multiple Linear Regression is one of the important regression algorithms which model the linear relationship between a single dependent continuous variable and more than one independent variable."

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ...+ \beta_p x_{ip} + \epsilon$$

Where, for i = n observations:

- Y_i = dependent variable
- X_i = explanatory variables
- β_0 = y-intercept (constant term)
- β_p = slope coefficients for each explanatory variable
- ϵ = the model's error term (also known as the residuals)

iii. Decision Tree Regression

Decision tree regression is a machine learning technique used for forecasting numerical values. Unlike decision trees for classification, which predict categorical labels, decision tree regression predicts continuous outcomes. Decision tree regression involves:

- Select relevant features for prediction, possibly through feature engineering.
- Tuning hyper-parameters like tree depth and leaf samples through techniques like cross-validation.
- Addressing overfitting with pruning or ensemble methods like Random Forests.
- Handling missing values without needing imputation.
- Offering good interpretability by tracing predictions from root to leaf.
- Being insensitive to feature scale, eliminating the need for data scaling.
- Capturing non-linear relationships between features and the target variable, beneficial for forecasting tasks.

iv. Random Forest Regression

Random Forest regression is an ensemble learning method that utilizes multiple decision trees to perform regression tasks. It's a powerful technique for forecasting, combining the simplicity and interpretability of decision trees with the robustness of ensemble learning.

Here's how Random Forest regression works:

- Utilizes bootstrapped sampling to build multiple decision trees from training data subsets.
- Implements random feature selection at each node to reduce the correlation between trees and prevent overfitting.
- Constructs decision trees with the bootstrapped sample and randomly selected features, usually without pruning.
- Aggregates predictions from all trees, typically averaging for regression tasks, yielding a more robust forecast.

v. Gradient Boosting Regression

Gradient Boosting regression is another powerful machine learning technique used for forecasting tasks. It belongs to the family of ensemble methods like Random Forest, but it builds a sequence of decision trees serially, with each tree aiming to correct the errors made by the previous ones. Here's how Gradient-boosting regression works:

- Starts by fitting an initial model, such as the mean of the target variable or a small decision tree.
- Calculates residuals between actual target values and predictions.
- Fits subsequent decision trees to the residuals to improve prediction accuracy.
- Optimizes parameters using gradient descent to minimize a loss function, with the learning rate controlling the step size.
- Combines predictions from all trees, focusing on correcting errors sequentially for a more accurate ensemble prediction.

Gradient Boosting regression advantages for forecasting:

- High predictive accuracy, often surpassing other methods, particularly in complex data scenarios.
- Capable of capturing non-linear relationships and interactions, making it suitable for complex forecasting tasks.
- Less prone to overfitting due to sequential tree building and optimization for residual errors.
- Provides feature importance measures, aiding in understanding influential features in the forecasting process, akin to Random Forests.

vi. Support Vector Machines Algorithm

Support Vector Machines (SVM) can also be used for regression tasks, including forecasting. When applied to regression, SVM is often referred to as Support Vector Regression (SVR). Support Vector Regression (SVR) for forecasting involves:

- Mapping input features into a higher-dimensional space using a kernel function to capture complex relationships.
- Finding a hyperplane in this space that maximizes the margin between it and the closest data points.
- Introducing a loss function that penalizes deviations from the hyperplane within an ε-insensitive tube, with regularization to control overfitting.
- Utilizing different kernel functions to capture non-linear relationships.

Advantages:

- Flexibility in capturing non-linear patterns.
- Robustness to outliers due to the ε-insensitive loss function.
- Control over model complexity via regularization.
- Typically results in global optimum solutions.

Considerations:

- Kernel selection and parameter tuning significantly impact performance.
- Training can be computationally intensive, requiring efficient implementations.
- Interpretability may be challenging compared to simpler methods like linear regression.

vii. K-Nearest Neighbour (KNN) Regression Algorithm

K-nearest neighbors (KNN) regression is a simple yet effective method for forecasting numerical values. Here is how it works:

- Utilizes feature vectors and target values in a multidimensional space.
- Calculates distances between new data points and all training points using a chosen metric.
- Select the K nearest neighbors based on calculated distances.
- Predicts the target value by averaging (or weighting) the target values of the K nearest neighbors.

Considerations:

- Choice of K impacts model flexibility and noise sensitivity.
- Selection of an appropriate distance metric depends on data characteristics.
- Feature scaling is crucial for balanced distance calculations.

- Predicting with KNN can be computationally intensive, especially for large datasets.
- Handling missing values can be done through imputation using neighbouring points.

viii. Lasso Regression Algorithm

Lasso Regression, also known as L1 regularization, is a linear regression technique used for forecasting that incorporates regularization to prevent overfitting and encourage sparsity in the model coefficients. Here's how Lasso Regression works:

Objective Function: Like traditional linear regression, Lasso Regression aims to minimize the difference between the predicted values and the actual target values. However, it adds a regularization term to the loss function, which penalizes the absolute values of the coefficients of the features. The objective function for Lasso Regression is given by:

Minimize $[(1/2n) \| y - Xw \|_2 2 + \alpha \| w \|_1]$

Where:

- y is the vector of target values
- X is the feature matrix
- w is the vector of coefficients (weights)
- n is the number of samples
- $\| . \|_2$ denotes the L2 norm (Euclidean norm)
- $\| . \|_1$ denotes the L1 norm
- α is the regularization parameter, also known as the Lasso regularization parameter. It controls the strength of regularization.

Lasso Regression advantages for forecasting:

- Automatically performs feature selection by setting less important coefficients to zero, enhancing model interpretability and potentially improving generalization.
- Prevents overfitting by penalizing coefficients' absolute values, resulting in smoother and more stable predictions.
- Handles multi-collinearity by selecting one feature from correlated groups and setting others' coefficients to zero.

Implementation:

Lasso Regression can be implemented using libraries like sci-kit-learn in Python, which offers efficient implementations, hyper-parameter tuning, and evaluation tools**.**

ix. Ridge Regression Algorithm

Ridge Regression, also known as Tikhonov regularization, is a linear regression technique used for forecasting that incorporates regularization to prevent overfitting. Here's how Ridge Regression works:

Objective Function: Similar to ordinary least squares (OLS) linear regression, Ridge Regression aims to minimize the difference between the predicted values and the actual target values. However, it adds a regularization term to the loss function, which penalizes the squared magnitudes of the coefficients of the features. The objective function for Ridge Regression is given by:

minimize$((1/2n) \| y - Xw \|_2 2 + \alpha \| w \|_2 2)$

where:

- y is the vector of target values.
- X is the feature matrix.
- W is the vector of coefficients (weights) to be learned.
- n is the number of samples.
- $\| . \|_2$ denotes the L2 norm (Euclidean norm).
- α is the regularization parameter, also known as the Ridge regularization parameter. It controls the strength of the regularization.

Ridge Regression advantages for forecasting:

- Prevents overfitting by penalizing large coefficients, promoting stability and better generalization.
- Handles multi-collinearity by reducing individual features' impact on predictions, mitigating collinearity's influence.
- Produces more stable solutions compared to OLS linear regression, especially with high-dimensional or multi-collinear datasets.

Implementation:

- Ridge Regression can be implemented using libraries like sci-kit-learn in Python, offering efficient implementations, hyper-parameter tuning, and evaluation tools.

x. Elastic Net Regression Algorithm

Elastic Net Regression is a linear regression technique used for forecasting that combines the penalties of both Lasso Regression (L1 regularization) and Ridge Regression (L2 regularization) to achieve a balance between the two. It is particularly useful when dealing with datasets with high-dimensional feature spaces and multi-collinearity. Here's how Elastic Net Regression works:

Objective Function: The objective function for Elastic Net Regression is a combination of the L1 and L2 regularization terms added to the standard least squares loss function. It aims to minimize the following expression:

minimize $((1/2n) \|y - Xw\|_2 2 + \alpha\rho \|w\|_1 + (\alpha(1-\rho)/2) \|w\|_2 2))$

where:

- y is the vector of target values.
- X is the feature matix
- w is the vector of coefficients (weights)
- n is the number of samples
- $\| . \|_2$ denotes the L2 norm (Euclidean norm)
- $\| . \|_1$ denotes the L1 norm
- α is the overall regularization parameter, controlling the strength of regularization.
- ρ is the mixing parameter, which determines the balance between L1 and L2 regularization penalties. When ρ = 1, it reduces to Lasso Regression, and when ρ = 0, it reduces to Ridge Regression.

Elastic Net Regression combines Lasso and Ridge Regression, offering benefits like handling multi-collinearity, robustness with correlated features, and flexible parameter tuning. To optimize forecasting performance, tune regularization parameters (α and ρ) through techniques like cross-validation. Python libraries like sci-kit-learn facilitate implementation and hyper-parameter tuning.

IV. SIMULATION AND RESULTS

A. About the Data set

We have considered data provided by the Kaggle community in the heading of Short-term electricity load forecasting (Panama case study). Data sources provide hourly records. The data composition is the following:

1. Historical electricity load, available on daily post-dispatch reports, from the grid operator (CND).
2. Historical weekly forecasts available on weekly pre-dispatch reports, both from CND.
3. Calendar information related to school periods, from Panama's Ministry of Education.
4. Calendar information related to holidays, from the "When on Earth?" website.
5. Weather variables, such as temperature, relative humidity, precipitation, and wind speed, for three main cities in Panama, from Earth data.

B. Data Pre-processing and evaluation method

This data was already processed and cleaned for load forecasting hence this part has been skipped. the original data sources provide the post-dispatch electricity load in individual Excel files daily and weekly pre-dispatch electricity load forecast data in individual Excel files every week, both with hourly granularity. holidays and school periods data is sparse, along with websites and pdf files. weather data is available on daily netcdf files. The labeling of the prediction variables has been done as follows:

Abbreviations	Meaning
nat_demand	National electricity load
T2M	Temperature at 2 meters
Dav	David City
QV2M	Relative humidity at 2 meters
TQL	Liquid precipitation
W2M	Wind speed at 2 meters
Toc	Tocumen, Panama City
San	Santiago city

Table 2: The table shows the meaning of the various prediction variables used in the forecasting**.**

C. Simulation Setup

We set-up the environment for simulation using the popular Jupyter Notebook, also known as IPython Notebook, is widely utilized for Python coding, particularly in data analysis, data science, and machine learning domains. Its user-friendly

interface allows for swift code execution and output review, facilitating an iterative approach essential for data analytics, hypothesis testing, and result documentation, akin to using a traditional notebook.

D. Numerical Results

We first visualize the data with box plots by having days of the week on the x-axis and energy consumption(MW) on the y-axis.

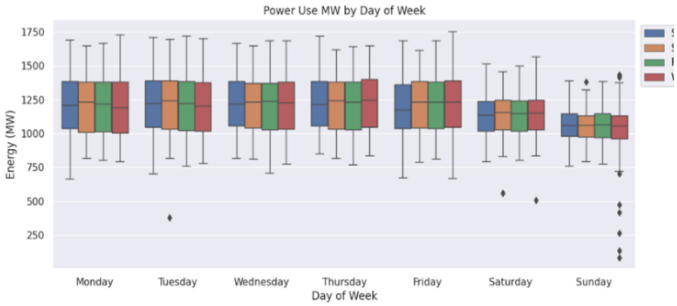

Fig.1: The boxplot also gives us the consumption of the power for the different seasons of the year. By visual inspection is evident that the consumption is lower on weekends like Saturdays and Sundays for different seasons of the year.

We also observe here some outliers (diamond-shaped elements). As outliers cause prediction inaccuracy and have several disadvantages we remove them using the IQR(Inter-Quartile method) and this is how it looks like after outlier removal (Refer to fig.2):

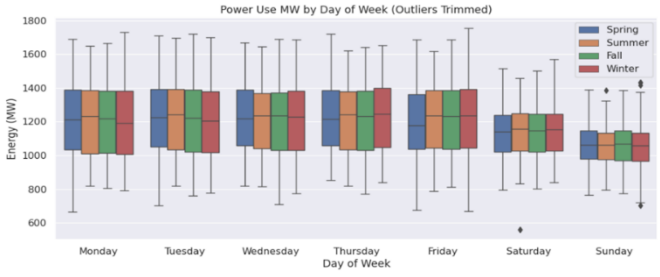

Fig.2: Boxplot after outlier removal. By visual inspection, we see that the outliers have been successfully removed.

Further, we import the Scikit-learn library. It is a Python library, offers various supervised and unsupervised learning algorithms, leveraging familiar tools like NumPy, pandas, and Matplotlib. Its functions cover regression (Linear and Logistic), classification (like K-Nearest Neighbors), clustering (including K-Means and K-Means++), model selection, and preprocessing (such as Min-Max Normalization).

Then we perform the various regression algorithms one by one on the data set and get the RMSE (Root Mean Squared Error) and R2(R-squared) score for each of them for comparison.

Fig. 3 shows the results of RMSE and R2 for regression methods namely, linear regression, Decision Tree, Random Forest, Gradient Boosting and SVR.

```
Model: Linear Regression
Root Mean Squared Error (RMSE): 170.87530204397436
R-squared (R2) Score: 0.1556602659077979

Model: Decision Tree
Root Mean Squared Error (RMSE): 149.07036969970065
R-squared (R2) Score: 0.3573991525645632

Model: Random Forest
Root Mean Squared Error (RMSE): 135.46520115735763
R-squared (R2) Score: 0.4693427166662475

Model: Gradient Boosting
Root Mean Squared Error (RMSE): 127.30877036108208
R-squared (R2) Score: 0.5313212275581627

Model: SVR
Root Mean Squared Error (RMSE): 165.65802749486647
R-squared (R2) Score: 0.20643298360725137
```

Fig.3: Results of RMSE and R2 for various regression methods (Part -1).

Fig. 4 shows the results of RMSE and R2 for regression methods namely, Lasso regression, Ridge Regression, and Elastic Net.

```
Model: Lasso Regression
Root Mean Squared Error (RMSE): 170.60616179784125
R-squared (R2) Score: 0.158317956484766

Model: Ridge Regression
Root Mean Squared Error (RMSE): 174.1004307127111
R-squared (R2) Score: 0.12348706814718546

Model: ElasticNet
Root Mean Squared Error (RMSE): 181.0937999437399
R-squared (R2) Score: 0.05165621935602005
```

Fig.4: Results of RMSE and R2 for various regression methods (Part -2).

On comparing manually and also by computing we find out that the lowest score for RMSE value is for the Gradient booting method which is around 127 and hence it becomes the best regression technique for the given set of data for electricity load forecasting. This is shown by computation in Fig. 5 below:

```
Best Hyperparameters: {'learning_rate': 0.2, 'max_depth': 4, 'n_estimators': 50}
Root Mean Squared Error (RMSE): 127.38845636734575
R-squared (R2) Score: 0.5307343264706192
```

Fig.5: Shows the computational result for the best regression method.

V. FUTURE WORK AND REMARKS

The proposed approach of employing various regression methods using Python, evaluated by the metrics RMSE and R-squared, offers valuable insights into electricity load forecasting. The thorough examination of regression techniques enhances our understanding of their effectiveness in this domain.

For future research, several promising avenues could be explored. Firstly, integrating advanced deep learning architectures such as Long Short-Term Memory (LSTM) networks [1] and transformer-based models [2] could potentially improve forecasting accuracy, especially in capturing long-term dependencies and complex patterns. Additionally, exploring hybrid models like PLCNet, which combines deep LSTM and CNN architectures in parallel, could provide further enhancements in accuracy and efficiency.

Furthermore, considering traditional time series models such as ARIMAX (Auto-Regressive Integrated Moving Average with Exogenous Variables) and ARIMA (Auto-Regressive Integrated Moving Average) [8] alongside machine learning approaches would offer a comprehensive comparison and understanding of their respective strengths and weaknesses in electricity load forecasting.

Overall, investigating these avenues would contribute to advancing the state-of-the-art in electricity load forecasting, potentially leading to more robust and accurate predictive models with practical applications in energy management and planning.

VI. CONCLUSION

Table 3, below shows the results of all the regression methods in a tabulated form.

Regression method	RMSE	R2 score
Linear regression	170.8753	0.1556
Decision Tree	149.0703	0.3573
Random Forest	135.4652	0.4693
Gradient Boosting	127.3087	0.5313
SVM	165.6580	0.2064
KNN	144.4757	0.3964
Lasso Regression	170.6061	0.1583
Ridge Regression	174.1004	0.1234
ElasticNet	181.0937	0.0516

Table 3: Shows the results of all the regression methods in a tabulated form.

In conclusion, our research on electricity load forecasting using various regression methods implemented in Python has yielded insightful results. Among the methods evaluated, Gradient Boosting emerged as the most effective, with a RMSE value of 127.3087 and an R-squared score of 0.5313. It significantly outperformed Random Forest and KNN, which followed with RMSE values of 135.4652 and 144.4757, and R-squared scores of 0.4693 and 0.3964, respectively.

Conversely, ElasticNet and Linear Regression exhibited comparatively poorer performance, indicating their limited suitability for this particular dataset, with RMSE values of 181.0937 and 170.8753, and R-squared scores of 0.0516 and 0.1556, respectively.

However, it's important to note that the choice of the best regression method is contingent upon various factors beyond just RMSE and R-squared scores. Other parameters, dataset characteristics, and specific forecasting requirements also play crucial roles in determining the optimal approach.

Thus, while Gradient Boosting shows promise as a robust method for electricity load forecasting in this study, further research should explore the performance of other advanced techniques, such as Long Short-Term Memory (LSTM) networks or transformer-based models, to enhance predictive accuracy and generalizability across diverse datasets and scenarios. Additionally, conducting sensitivity analyses on different hyper-parameters and feature engineering techniques could provide deeper insights into the factors influencing model performance and guide future advancements in this field.

REFERENCES

[1] Islam, B. ul, & Ahmed, S. F. (2022). Short-Term Electrical Load Demand Forecasting Based on LSTM and RNN Deep Neural Networks. Hindawi Mathematical Problems in Engineering, 2022, Article ID 2316474, 10 pages.

[2] L'Heureux, A.; Grolinger, K.; Capretz, M.A.M. Transformer-Based Model for Electrical Load Forecasting. Energies 2022, 15, 4993

[3] Jawad, M., Nadeem, M. S. A., Shim, S. O., Khan, I. R., Shaheen, A., Habib, N., Hussain, L., & Aziz, W. (2020). Machine Learning Based Cost Effective Electricity Load Forecasting Model Using Correlated Meteorological Parameters. IEEE Access, 8, 146847-146864. DOI: 10.1109/ACCESS.2020.3014086.

[4] BEHNAM FARSI, MANAR AMAYRI, NIZAR BOUGUILA, (Senior Member, IEEE), AND URSULA EICKER, Concordia Institute for Information Systems Engineering(CIISE), Concordia University, Montreal, QC H3G 1M8, Canada, 2G-SCOP Lab, Grenoble Institute of Technology, 38185 Grenoble, France

[5] Phyo, P. P., & Jeenanunta, C. (2021). Daily Load Forecasting Based on a Combination of Classification and Regression Tree and Deep Belief Network. IEEE Access, 9, 152242-152239. DOI: 10.1109/ACCESS.2021.3127211

[6] Sarhani, M., & El Afia, A. (2015). Electric load forecasting using a hybrid machine learning approach incorporating feature selection. In Proceedings of the International Conference on Big Data Cloud and Applications (pp. 1-7). Tetuan, Morocco: Mohammed-V University Rabat.

[7] "A Research on Electric Load Forecasting Factors Effecting and Methods Involved" International Journal of Innovative Technology and Exploring Engineering (IJITEE), Volume-8 Issue-12, in October 2019, R. Hariharan and More Kranthi Kumar.

[8] Shilpa. G. N, Dr. G. S. Sheshadri, "Electrical Load Forecasting Using Time Series Analysis," International Journal of Engineering Research and Development, Vol. 13, Issue 7, pp. 75-79, July 2017.
[9] Taık, A., & Cherkaoui, S. (Year). Electrical Load Forecasting Using Edge Computing and Federated Learning. INTERLAB, Engineering Faculty, Université de Sherbrooke, Canada.
[10] Cao, Q. (2022). A Survey of Electric Load Forecasting Algorithm Models. Highlights in Science, Engineering and Technology, 9, 471-483.