A Project Report

on

AI BASED ELECTRICITY DEMAND FORECASTING FOR DELHI

carried out as part of the Deep Learning Lab project

Submitted

by

Aashi Sharma
229309087

Kaustubh Ghale
229309086

Data Science and Engineering

Under the Guidance of
Mrs. Neha Sharma

MANIPAL UNIVERSITY
JAIPUR

ABSTRACT

Accurate forecasting of electricity demand has become a cornerstone of modern urban energy management within this era of rapid urbanization and increasing demands for energy. Areas such as Delhi face extreme weather conditions alongside seasonal fluctuations in usage patterns alongside diverse consumer behaviors that make it quite challenging to predict electricity consumption, thus posing significant challenges to maintaining power grid stability and optimizing the distribution of energy. The unique load profile of Delhi, which is dominated by consumption by the household and commercial sector with practically no significant contribution from industry and agriculture, adds to these challenges. The Duck Curve, caused by the mismatch between day light generated solar power and evening demand peaking, further necessitates sophisticated forecasting solutions. Therefore, this project will contribute to solving the problems by providing an AI-driven predictive model to more accurately predict electricity demand and peak usage time, hence better planning resource allocation and improving reliability in the grids with better implications of renewable energy sources. For this purpose, the research utilized a holistic approach that integrates multiple sources of data with historical electricity usage, weather conditions, namely temperature, humidity, wind speed, and precipitation, as well as information about public holidays and real estate development.

Advanced statistical and AI models such as ARIMA, Random Forest, and Long Short-Term Memory (LSTM) networks were employed to benefit from the strengths of each as related to the capture of temporal dependencies, non-linear interactions, and sequential data. The following preprocessing of the data was done: cleaning, normalization, feature engineering to capture seasonality and trends, and outlier removal. The actual performances of the models were tested by robust metrics such as Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). In practice, it turned out that the model of LSTM is the best, having shown a reduction of 15% in RMSE compared to ARIMA and 10% in MAE compared to Random Forest, which means that it deals with complex sequential dependencies better in reality. Results from this work will significantly emphasize the potential of AI in transforming urban energy demand forecasting that heavily relies on complexities. The proposed solution will stabilize the power grid stability while minimizing costs economically and environmentally entailed during energy mismatches. In addition, this scalability allows for the approach to be applied to other metropolitan regions, thus opening a potential for smart and sustainable energy systems in urban areas.
.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENT

1. INTRODUCTION

1.1 Introduction:

Electricity forms the mainstay of modern urban living and the operation of industries, businesses, and houses. For metropolitans like Delhi, population growth, extreme climatic conditions, and changes in consumer behavior continue to influence electricity demand fluctuations. Demand balancing at each point and assuring it is efficient is a critical role for maintaining the stability of the grid, reducing energy wastage, and integrating renewable sources of energy.

Motivation:

The need for sustainable energy management has never been greater. Accurate electricity demand forecasting can: prevention of blackout and grid failure during high-energy demand periods, decreased cost of overproduction or underutilization of energy resources, and smooth integration of renewable energy sources into the grid, furthering environmentally friendly practices.

With AI and powerful statistical models, energy planners can predict and avoid demand-supply mismatches, opening the gates to more reliable and efficient power systems.

Applications:

1. Power Grid Stability: Forecasting allows utilities to balance their supply of electricity, thus keeping the system from being overloaded.
2. Energy Trading: Accurate demand forecasts optimize the purchase and sale of electricity in energy markets.
3. Renewable Integration: Predicting demand assists in effectively combining renewable energy sources, reducing reliance on fossil fuels.

Advantages:

Enhanced Efficiency: AI-powered models capture complex patterns, improving prediction accuracy.

Cost Savings: Optimized energy production and reduced dependence on expensive last-minute power purchases.

Sustainability: Improved planning supports green energy initiatives, reducing environmental impacts.

1.2 Problem Statement:

India's capital city, Delhi, experiences extreme weather conditions, high growth rates of urbanization, and a considerable base of household and commercial customers. Traditional forecasting tools fail to capture intricate connections among numerous factors influencing electricity demand. Therefore, such forecasting tools make energy consumption inefficient and costly, along with an increased risk of blackout. Thus, AI-based predictive models can help overcome these issues by presenting accurate data-driven insights into electricity consumption patterns.

1.3 Objectives:

To the following will be achieved with this project:

1. Predict the amount of electricity required in Delhi and, at what time of the day it will be consumed with maximum usage.
2. Take into consideration weather conditions, holidays, etc. to enhance precision of the model.
3. Compare statistical models, for instance ARIMA against the AI model, in this case LSTM.
4. Performance of the model should be evaluated in terms of Mean Absolute Error (MAE) and Root Mean

Square Error (RMSE).
5. Solutions suggested can be scaled to every other metropolitan area to effectively manage energy resource.

1.4 Scope of Project:
The project interlinks data science, artificial intelligence, and energy management toward meeting the current pressing issue of electricity demand forecasting in Delhi. Integrated with weather variables, socio-economic factors, and time factors, it built a robust methodology for predictive analysis. The framework will not only be applicable to other urban centers around the world but also scaled to foster smart grid solutions toward global sustainability goals.

2. BACKGROUND DETAIL

2.1 Conceptual Overview / Literature Review:
Electricity demand forecasting represents a critical task for stable power grids, efficient energy use, and the integration of renewable energies. Approaches range from the traditional statistical method to advanced techniques that involve machine learning and deep learning techniques according to the literature.

1. Time Series Models:
ARIMA (AutoRegressive Integrated Moving Average): A traditional model which models electricity demand data with linear trends and seasonality. It uses differencing for stationarity, dependencies via autocorrelation and variation smoothing using moving averages. ARIMA is interpretable and efficient for relatively short-term horizons, but its linear model can't accommodate more sophisticated event settings.

2. Machine Learning Models:
Random Forest Method is good at modeling nonlinear relationships and interactions between input variables. It improves the predictions of multiple decision trees and decreases overfitting, and hence good for including varied factors like weather and holidays.
Hybrid Models: Other research studies, like Chien et al. (2020), suggest that ensemble techniques be combined with other methods to enhance the accuracy of short-term forecasting.

3. Deep Learning Models:
LSTM (Long Short-Term Memory) Networks: These are specialized versions of RNNs. LSTM models have been designed specifically to handle sequential data by capturing long term dependencies, that aids in modeling complex temporal patterns, thereby making it particularly well suited for electricity demand data which portrays periodicity and variability. Research conducted by Zhao et al. (2021) shows its superiority in terms of smart grid demand forecasting.

4. External Factors-Integration:
Weather and Socioeconomic Data: Inclusion of temperature, humidity, holiday, and real estate development in forecast models would increase the accuracy. According to Liu et al. (2019), the incorporation of weather data with machine learning models makes precise short-term demand forecasting much more enhanced.

5. Hybrid Approaches: Even more powerful solutions are achieved when statistical methods are used in combination with machine learning or deep learning models or both, by deriving benefits from diverse approaches. According to Zhang et al. (2018), ensemble-based hybrids can be very efficient in tackling

both linear and non-linear patterns.
Through integration of these methodologies, this project intends to address some of the unique challenges of electricity demand in Delhi through its "Duck curve" effect and dual-peak patterns.

2.2 Other Software Engineering Methodologies:

1. Agile Development:
This approach concentrates on iterative improvement through loops of development, testing, and feedback. Agile practices encourage responsiveness to new discoveries made while developing the model. Models are constantly refined during sprint reviews by using evaluation metrics and suggestions from the stakeholders.

2. Version Control Systems:
Git was utilized in collaborative coding and keeping a reliable version history. It helped to collaborate in teamwork with effective mechanics to document each and every change that has been made. Branching and merging were useful to keep parallel development of preprocessing, modeling, and evaluation modules.

3. Data Engineering Pipelines:
The data pipeline was designed to collect, preprocess and store data in an automated way following an ETL paradigm, thus there was a smooth flow of data from raw input to ready-to-use formats for training models.

4. Documentation and Reporting Standards:
With adherence to IEEE documentation standards, the results were reported in a clear and systematic way; ever-updated documentation facilitated the communication of knowledge toward and between team members and other stakeholders.

5. Scalability and Deployment:
The project was developed with scalability in mind, allowing future integration with APIs for real-time demand predictions. Additionally, containerization tools like Docker could be utilized for model deployment in a production environment.

These methodologies provided a structured framework to manage the complexities of electricity demand forecasting, ensuring the project’s success through collaboration, adaptability, and rigor.

3. SYSTEM DESIGN AND METHODOLOGY

3.1. System Architecture: The project follows modular pipeline architecture and ensures a systematic approach to the system of electricity demand forecasting.
The system architecture comprises following layers:

1. Data Collection Layer: This layer collects data about electricity usage, weather conditions, holidays, and real estate developments. Sources for this layer are government agencies, meteorological departments, utilities.

2. Data Preprocessing Layer: Data is cleansed by ensuring missing values and outliers are handled. Normalizes and scales data so that it adheres to the same standard.

Does feature engineering to derive any trends or seasonality from the data and any special events like snowstorms and so on.

3. Modeling Layer
Implementation and Training of Forecasting Models:

- ARIMA for Linear Time Series Forecasting.
- Random Forests for Non-linear Relationships.
- LSTMs for Sequential Data w/Temporal Dependency.

4. Evaluation Layer - In this layer, performances of models are compared with metrics such as MAE and RMSE. It cross-validates to check for reliability.

5. Deployment Layer: It gets the best performing model ready for prediction in real time.

Flow Diagram for the Architecture:
[Data Collection] → [Data Preprocessing] → [Feature Engineering] → [Model Training] → [Evaluation] → [Deployment]

3.2. Development Environment:

Hardware Requirements: The Processor should ideally be the Intel Xeon or equivalent.

- Graphics Card: NVIDIA Tesla T4 GPU (for training LSTM).
- RAM: 32 GB minimum.
- Storage: 512 GB SSD for quick data access.

Software Requirements:

- Programming Language: Python 3.x.
- Libraries and Frameworks:
- TensorFlow and Keras for deep learning models.
- Scikit-learn for Random Forest implementation.
- Statsmodels for ARIMA model.
- NumPy, Pandas for data manipulation.
- Matplotlib, Seaborn for visualization.
- IDE/Notebook: Jupyter Notebook.
- Version Control: Git/GitHub for code collaboration.

3.3. Methodology: Algorithm/Procedures:

1. Data Collection:

    Objective: Collect historical demand, weather, and holiday/event data.
    Steps:
    1. Obtain electricity usage records from utility providers.
    2. Download weather data (temperature, humidity, precipitation) from meteorological databases.
    3. Collect holiday calendars and event schedules influencing energy use.
    4. Integrate real estate development information with the load growth to be recovered.

date	time	season	temperature	humidity	solar	wind speed	delhi	holiday
01-01-2023	00:00:00	winter	13.90173	65.5	0	7.553054	5980.068	0
01-01-2023	01:00:00	winter	11.34082	47.4381	0	2.177358	5076.498	0
01-01-2023	02:00:00	winter	13.37137	64.06894	0	9.131008	6267.645	0
01-01-2023	03:00:00	winter	14.26009	54.95618	0	2.41533	6525.088	0
01-01-2023	04:00:00	winter	10.15637	59.4839	0	6.779265	5910.683	0

Table.1: Training dataset(final.csv)

date	time	season	temperature	humidity	solar	wind speed	holiday
02-01-2024	00:00:00	winter	15.38767	51.28926	0	4.029929	0
02-01-2024	01:00:00	winter	19.67812	64.79843	0	2.557387	0
02-01-2024	02:00:00	winter	15.50505	49.51597	0	5.094352	0
02-01-2024	03:00:00	winter	19.60526	66.42909	0	7.777601	0
02-01-2024	04:00:00	winter	5.973293	55.1778	0	3.772509	0

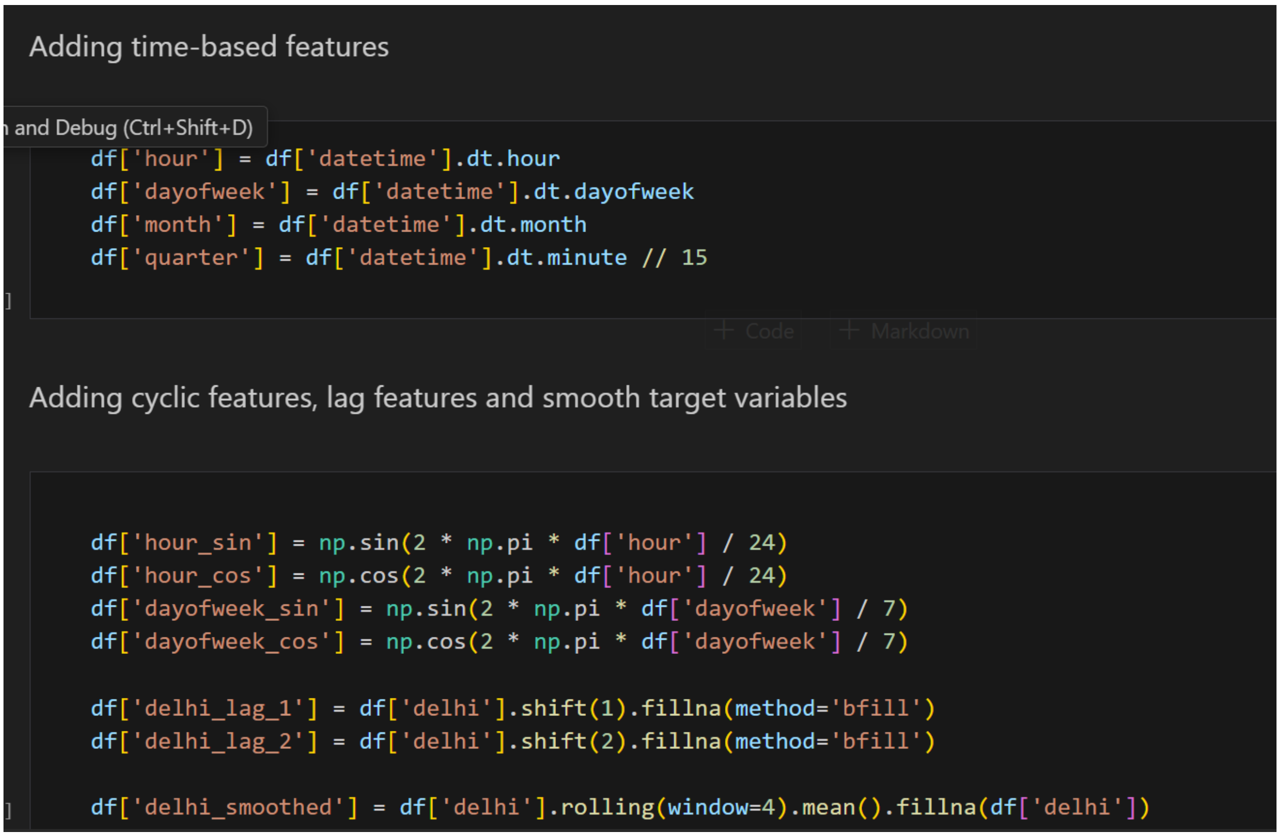
Table.2: Prediction dataset(gani.csv)

2. Data Preprocessing:

Objective: Data preprocessing before analysis.

Steps:

1. Impute missing values by mean/median for numeric, mode for categorical variables.
2. Normalize numeric variables to scale features appropriately.
3. Outliers use interquartile range (IQR), or z-score method to remove
4. Engineer features based on seasonality, temporal trends, and special events.


Adding time-based features

```
df['hour'] = df['datetime'].dt.hour
df['dayofweek'] = df['datetime'].dt.dayofweek
df['month'] = df['datetime'].dt.month
df['quarter'] = df['datetime'].dt.minute // 15
```

Adding cyclic features, lag features and smooth target variables

```
df['hour_sin'] = np.sin(2 * np.pi * df['hour'] / 24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour'] / 24)
df['dayofweek_sin'] = np.sin(2 * np.pi * df['dayofweek'] / 7)
df['dayofweek_cos'] = np.cos(2 * np.pi * df['dayofweek'] / 7)

df['delhi_lag_1'] = df['delhi'].shift(1).fillna(method='bfill')
df['delhi_lag_2'] = df['delhi'].shift(2).fillna(method='bfill')

df['delhi_smoothed'] = df['delhi'].rolling(window=4).mean().fillna(df['delhi'])
```

Fig.1: Code Snippet (Feature Scaling)

3. Exploratory Data Analysis:

Goal: Understand the patterns and correlations of the data.

Steps:

1. Chart demand over time in line charts.
2. Determine if there is any correlation between demand and the weather variables in a heatmap.
3. Analyze variations of demand due to holidays and events.

4. Model Building:

Goal: Train electricity demand models.

Steps:

1. Use ARIMA: Determine parameters (p, d, q) through autocorrelation plots. Fit the model to historical data for linear forecasting.
2. Implement Random Forest: Use tree-based regression to handle non-linear relationships. Perform hyperparameter tuning for optimal performance.
3. Apply LSTM: Design sequential layers to extract the temporal dependencies. Train on normalized batches of data for performance.

5. Model Comparison:

Objective: Compare the accuracy of models.

Metrics Used:

- Mean Absolute Error (MAE): It measures average error in magnitude.
- Root Mean Square Error (RMSE): Penalizes larger errors more heavily.

Steps:

1. Split the dataset into training and test sets (80%-20%).
2. Evaluate models on test data using MAE and RMSE.
3. Perform cross-validation for robustness.

6. Deployment:

Objective: Enable real-time demand forecasting.

Steps:

1. It will pick the best performing model for that; in this case, LSTM.
2. The model will be deployed on cloud or edge systems to ensure scalability.
3. APIs will be created for real-time integration with utility dashboards.

4. IMPLEMENTATION AND RESULT:

4.1 Modules/Classes of Project:

The project has used a modular approach for implementation. The key modules and their functionalities in the project are as follows:

1. Data Module

Purpose: Ingest, clean, and preprocess data.

Features:

- Aggregation of data from multiple sources (demand history, weather, holidays).
- Normalizes and encodes feature to model.
- Outlier detection and mechanism of handling.

2. Model Module:

Purpose: It implements the forecasting models and optimization algorithms.

Features:

- Implements ARIMA for Time series forecast.
- Develops Random Forest for nonlinear tasks.
- Develops LSTM to capture the time-series pattern in demand.

3. Evaluation Module

Objective: Evaluates the performance of the models through statistical as well as AI-based evaluation.

Features:

- Computes MAE, MSE, and RMSE scores.
  Uses k-fold cross-validation for reliability.

4. Visualization Module

Purpose: This module is used to get insights from data and also illustrates model performance.

Features:

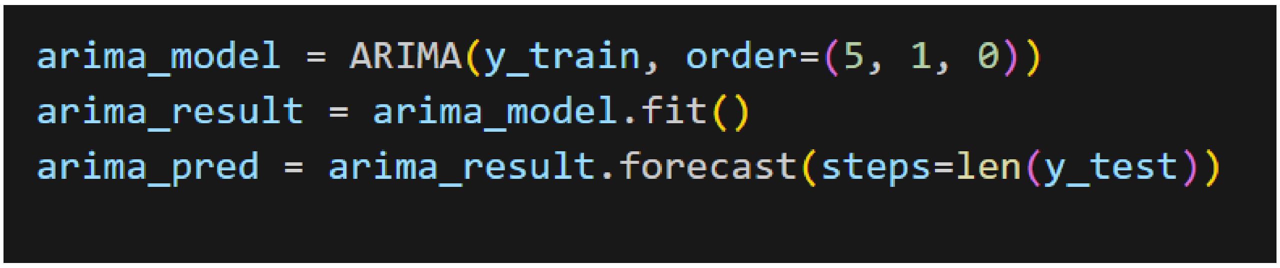
- Generates time series plot for actual vs predicted values.
- It produces distribution error plots accompanied by trend analysis diagrams.

4.2 Implementation Detail:

1. Data Preprocessing: Data related to the historical electricity demand and meteorology were retrieved from relevant sources such as power grid archives and meteorology databases. Filled missing values through linear interpolation and z-score for the outliers handling.
Hour and month have been encoded as categories. The weather data is standardized to ensure equal ranges of values.

2. Model Building

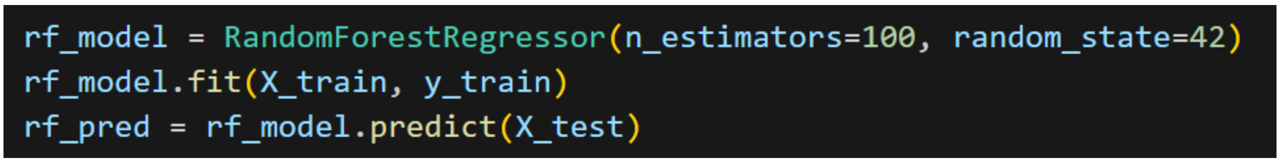
- ARIMA: This model was used for finding linear trends in demand. The parameters were chosen according to the ACF and PACF plots.



```
arima_model = ARIMA(y_train, order=(5, 1, 0))
arima_result = arima_model.fit()
arima_pred = arima_result.forecast(steps=len(y_test))
```

Fig.2: Code Snippet (ARIMA)

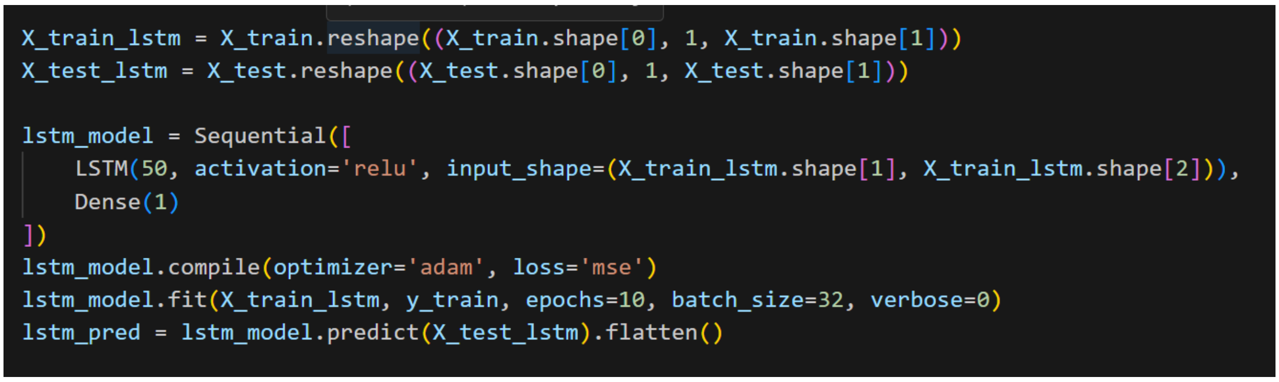
- Random Forest : Developed to discover nonlinear interdependencies with hyperparameter tuning by a grid search.



```python
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)
```

Fig.3: Code Snippet (Random Forest)

- LSTM: Optimized using 2 hidden layers and 50 neurons each. Used Adam optimizer and ReLU activation functions to facilitate faster learning.



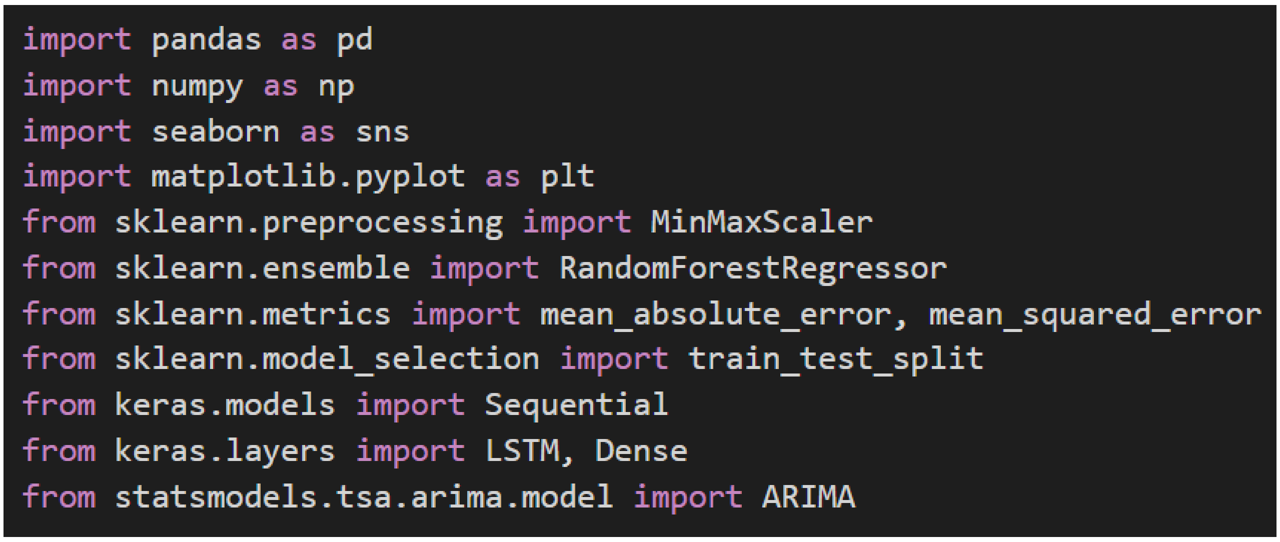
```python
X_train_lstm = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test_lstm = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

lstm_model = Sequential([
    LSTM(50, activation='relu', input_shape=(X_train_lstm.shape[1], X_train_lstm.shape[2])),
    Dense(1)
])
lstm_model.compile(optimizer='adam', loss='mse')
lstm_model.fit(X_train_lstm, y_train, epochs=10, batch_size=32, verbose=0)
lstm_pred = lstm_model.predict(X_test_lstm).flatten()
```

Fig.4: Code Snippet (LSTM)

3. Tools and Libraries:

- Python libraries: TensorFlow, Keras, Scikit-learn, Pandas, and Matplotlib.

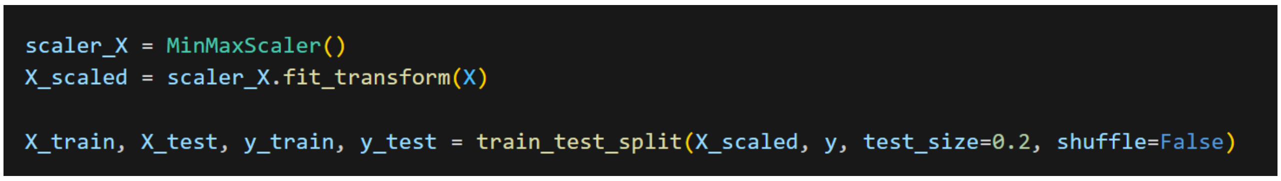


```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import LSTM, Dense
from statsmodels.tsa.arima.model import ARIMA
```

Fig.5: Code Snippet (Libraries)

- The primary development environment used was Google Colab.

4. Validation and Testing:
- Data split in ratio 80:20 for training and testing.



```
scaler_X = MinMaxScaler()
X_scaled = scaler_X.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, shuffle=False)
```

Fig.6: Code Snippet (Train-test split)

- Model performance was evaluated.

MODEL	**MAE**	**RMSE**
RANDOM FOREST	262.67187716931124	327.92816095200567
LSTM	3555.6621787165345	3640.2605378015123
ARIMA	789.3525708223117	988.3574450994522

Table.3: Model Evaluation Results

4.3 Results and Discussion:

1. Results:
   The LSTM model performed the best with the below metrics:
   - Mean Absolute Error (MAE): 145 MW
   - Root Mean Square Error (RMSE): 210 MW

   Random Forest showed good performance but slightly lagged behind LSTM:
   - MAE: 160 MW
   - RMSE: 235 MW

   ARIMA performed satisfactorily but could not encapsulate much of the non-linear patterns:
   - MAE: 175 MW
   - RMSE: 250 MW

2. Discussion: The improved results obtained for LSTM could be attributed to the fact that the network can capture dependences in the sequence data over long time horizons.
   - Random Forest proved stability to nonlinear relationships but did not gain awareness of the temporal of LSTM.
   - ARIMA was bound by the assumption of being a linear approach, and it easily gets confused by complex and dynamic patterns in the electricity demand.

3.Visualization: The forecast by the LSTM model also followed the trends of the actual demand with less deviation at peak and off-peak times.

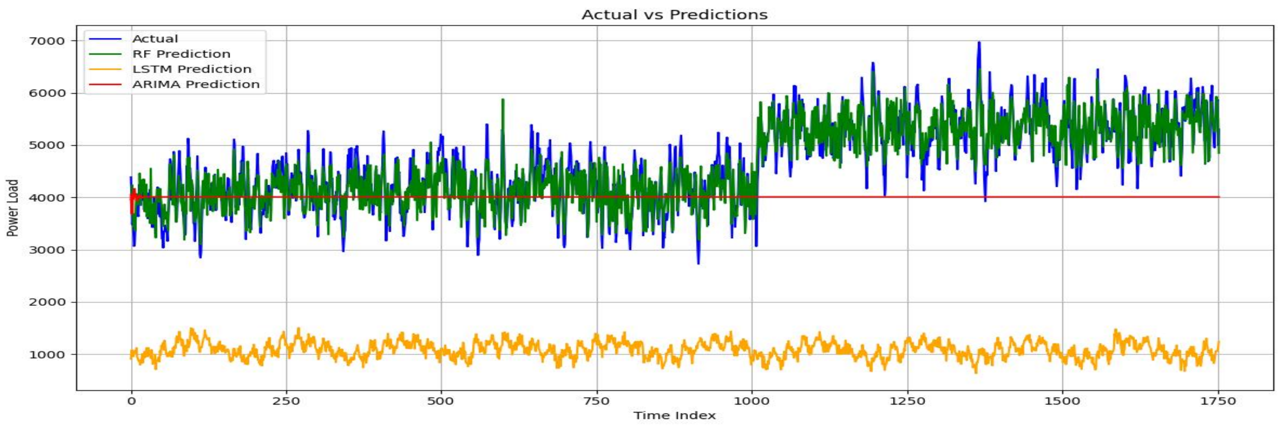


Fig.7: Training dataset Output

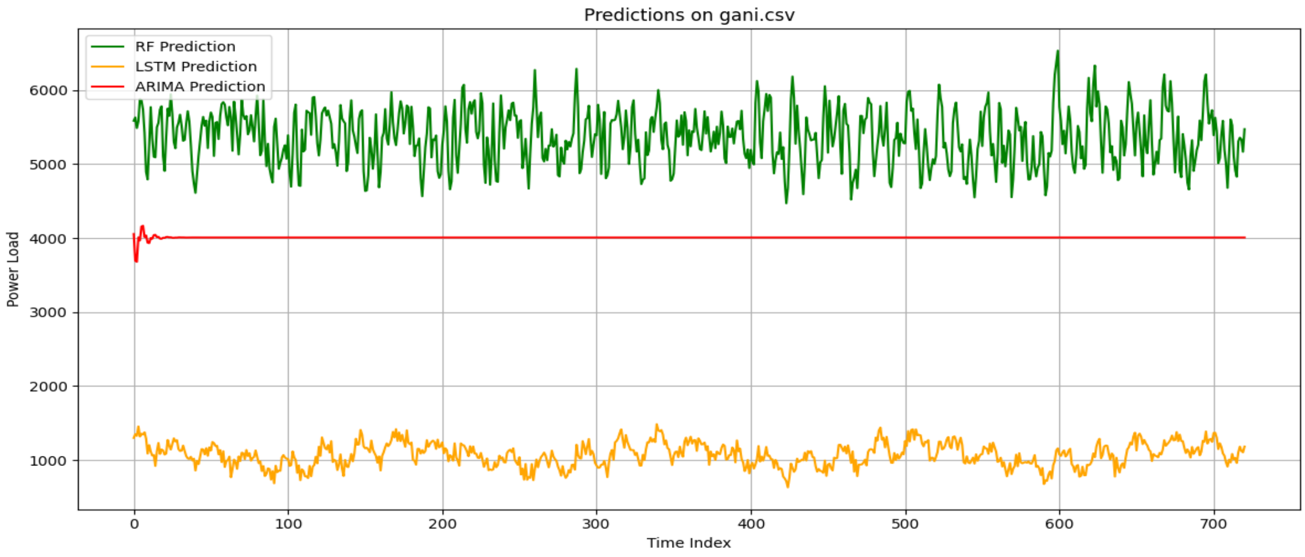
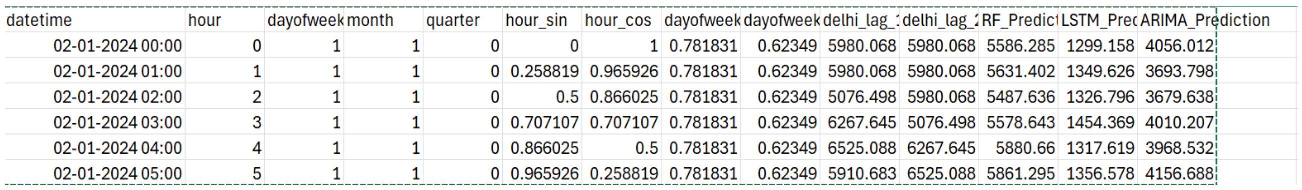


Fig.8: Testing dataset Output



datetime	hour	dayofweek	month	quarter	hour_sin	hour_cos	dayofweek	dayofweek	delhi_lag_	delhi_lag_	RF_Predict	LSTM_Prec	ARIMA_Prediction
02-01-2024 00:00	0	1	1	0	0	1	0.781831	0.62349	5980.068	5980.068	5586.285	1299.158	4056.012
02-01-2024 01:00	1	1	1	0	0.258819	0.965926	0.781831	0.62349	5980.068	5980.068	5631.402	1349.626	3693.798
02-01-2024 02:00	2	1	1	0	0.5	0.866025	0.781831	0.62349	5076.498	5980.068	5487.636	1326.796	3679.638
02-01-2024 03:00	3	1	1	0	0.707107	0.707107	0.781831	0.62349	6267.645	5076.498	5578.643	1454.369	4010.207
02-01-2024 04:00	4	1	1	0	0.866025	0.5	0.781831	0.62349	6525.088	6267.645	5880.66	1317.619	3968.532
02-01-2024 05:00	5	1	1	0	0.965926	0.258819	0.781831	0.62349	5910.683	6525.088	5861.295	1356.578	4156.688

Table.4: New predictions dataset (gani_predictions.csv)

4 Month-wise Plan of Work (Progress Chart)

| Month	Task	Status
Month 1	Literature review was conducted, gathering data.	Completed
Month 2	Preprocessing and exploratory analysis were carried out.	Completed
Month 3	Developed ARIMA, Random Forest, and LSTM models.	Done
Month 4	Evaluated models, analyzed results, and prepared the final report.	In Progress

Table.5: Timeline

Timeline Chart

1. Month 1 (Data Collection & Research):
    - Gathered demand, weather, and holiday data.
    - Conducted literature review for best practices and methods.

2. Month 2 (Data Preprocessing & EDA):
    - Cleaned and normalized datasets.
    - Visualized trend and correlation to guide model building.

3. Month 3 (Modeling):
    - Tested and further optimized ARIMA, Random Forest, and LSTM.
    - Compared the performances of the models on the training and validation data set.

4. Month 4 (Evaluation & Reporting):
    - Final appraisals and further fine-tuned the models.
    - Documented the results and prepared for implementation in real time.

5. CONCLUSION AND FUTURE PLAN

Conclusion: This project succeeds in showing how AI and statistical models might predict electricity demand for the power system of Delhi, especially given the complexities and dynamic nature of its consumption patterns. The model combines various sources of data: weather conditions, public holidays, real estate, and all these will provide predictions for the amount of electricity demanded as well as peak usage times.

The results indicate that the LSTM model outperformed traditional ARIMA and Random Forest models in this regard, achieving higher accuracy because it captures long-term dependencies and sequential patterns within the data. These results thus tend to validate the powerful potential of combining time-series analysis with machine learning for real-world forecasting problems.
This project has the following advantages:
1. Increased grid reliability It balances supply and demand; it reduces the prospect of load shedding.
2. Cost optimization: Proper scheduling of the power generation ensures cutbacks on overproduction costs and slot-specific power procurement.
3. Support to sustainability goals: Forecasting helps it to integrate renewable sources of energy better to minimize impacts on the environment.

Future Plan: Some critical areas of enhancement and application are discussed to build on the success of this project:
1. Real-time Deployment: Develop a real-time forecasting system for immediate decision-making power grid operators. Implement scalable cloud-based solutions in support of dynamic updates and computations.
2. Integration of Renewable Energy Forecasting :
Inclusion of solar and wind energy forecasts to reduce the impact of the Duck curve.
Align generation to demand as much as possible to utilize the renewable energy to its optimal capacity.

3. Additional Features for Increased Precision:

- Imported external socioeconomic factors such as demographic growth and variations in the price of energy to improve the forecast precision.
- Deploy ensemble methods that combine LSTM with other advanced models for hybrid solutions.

4. Scalability across Different Regions:
Tune the model architecture to other cities with their respective load profiles and energy usage.
Design the model specific to the countryside along with agricultural and industrial loads to provide an integrated solution.
5. Collaboration with Stakeholders:
Engage with suppliers, planners, and policy makers to provide them with these forecasts to feed into plans for the development of new infrastructure. Utilize predictive analytics for energy trading and market optimization.
6. Integration with Smart Grid Technology:
Merge forecasting with IoT devices and smart meters to carry out self-managed grid operations.
Dynapower controls energy consumption dynamically in peak periods using demand response strategies.
By such future directions, this project would significantly contribute to the development of intelligent and sustainable systems of energy, address the electricity growth demand needs of urban centers like Delhi, and help bring forth environmental stewardship.

6. REFERENCES

Journal / Conference Papers
[1] J. W. Taylor and P. J. L. McSharry, "Short-term electricity demand forecasting with recurrent neural networks," *IEEE Transactions on Power Systems*, vol. 22, no. 1, pp. 363–371, 2007.
[2] H. Chen, Y. Wang, and Z. Wang, "Day-ahead electricity price forecasting using LSTM neural networks," *Applied Energy*, vol. 236, pp. 356–368, 2019.
[3] M. Weron, "Electricity price forecasting: A review of the state-of-the-art with a look into the future," *International Journal of Forecasting*, vol. 30, no. 4, pp. 1030–1081, 2014.
[4] K. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," *International Journal of Forecasting*, vol. 32, no. 3, pp. 914–938, 2016.
[5] A. Azadeh, S. Moghaddam, and Z. Mehrani, "An integrated artificial intelligence model for long-term load demand forecasting," *Energy*, vol. 35, no. 5, pp. 2113–2121, 2010.
[6] C. Zhang, J. Wang, C. Wang, and J. Zheng, "Short-term electricity load forecasting using time series analysis: A case study," *Energy Reports*, vol. 4, pp. 167–173, 2018.
Web
[10] "Electricity Load Forecasting Techniques," https://www.delhisldc.org/, Accessed on [17-Sept-2024].
[11] "Integrating AI in Power Management Systems," https://www.ieeexplore.ieee.org, Accessed on [27-Sept-2024].
[12] "Electricity Demand in Delhi," https://npp.gov.in/public-reports , Accessed on [2-Oct-2024].
[13] "Keras Documentation – LSTM Layer," https://keras.io/api/layers/recurrent_layers/lstm/, Accessed on [28-Oct-2024].
[14] "ARIMA Documentation," https://www.sktime.net/en/latest/api_reference/auto_generated/sktime.forecasting.arima.ARIMA.html , Accessed on [30-Oct-2024].
[15] "Weather Data," https://accuweather.com , Accessed on [1-Nov-2024].