**ELECTRICITY PRICES PREDECTION USING**

**MECHINE LEARNING**

**BATCH MEMBER**

S.Dinesh kumar(411521104030)

J.Deepak (411521104017)

S.Karthi (411521104051)

K.Deepan chakkaravarthi(411521104020)

**PROJECT TITLE:ELECTRICITY PRICES PREDECTION**

**PHASE 3: DEVELOPMENT PART 1**



**INTRODUCTION:**

The price of electricity depends on many factors. Predicting the price of electricity helps many businesses understand how much electricity they have to pay each year. The Electricity Price Prediction task is based on a case study where you need to predict the daily price of electricity based on the daily consumption of heavy machinery used by businesses. So if you want to learn how to predict the price of electricity, then this article is for you. In this article, I will walk you through the task of electricity price prediction with machine learning using Python.

The ability to accurately forecast electricity prices holds immense value. By leveraging advanced algorithms and historical data, these powerful technologies unlock the potential to predict price fluctuations and optimize energy consumption. From enabling efficient resource allocation to aiding market participants in making informed decisions, the integration of data science and machine learning revolutionizes the electricity industry with unprecedented insights and foresight.

we delve into the realm of data science to forecast electricity prices with utmost precision. With the transformative power of advanced algorithms and predictive modeling, we aim to uncover underlying patterns and trends that shape the volatile realm of electricity markets. Join us as we explore the intricacies of this crucial field and reveal the potential of data-driven insights in anticipating price fluctuations.

**Importance of electricity price prediction**

Accurate electricity price prediction is vital for energy companies and consumers to make informed decisions. Optimizing data loading and processing ensures that machine learning algorithms can analyze large datasets efficiently. By implementing these techniques, we can enhance the accuracy of electricity price predictions, leading to cost savings, efficient resource planning, and a more sustainable energy market.

**Challenges in data loading and processing**

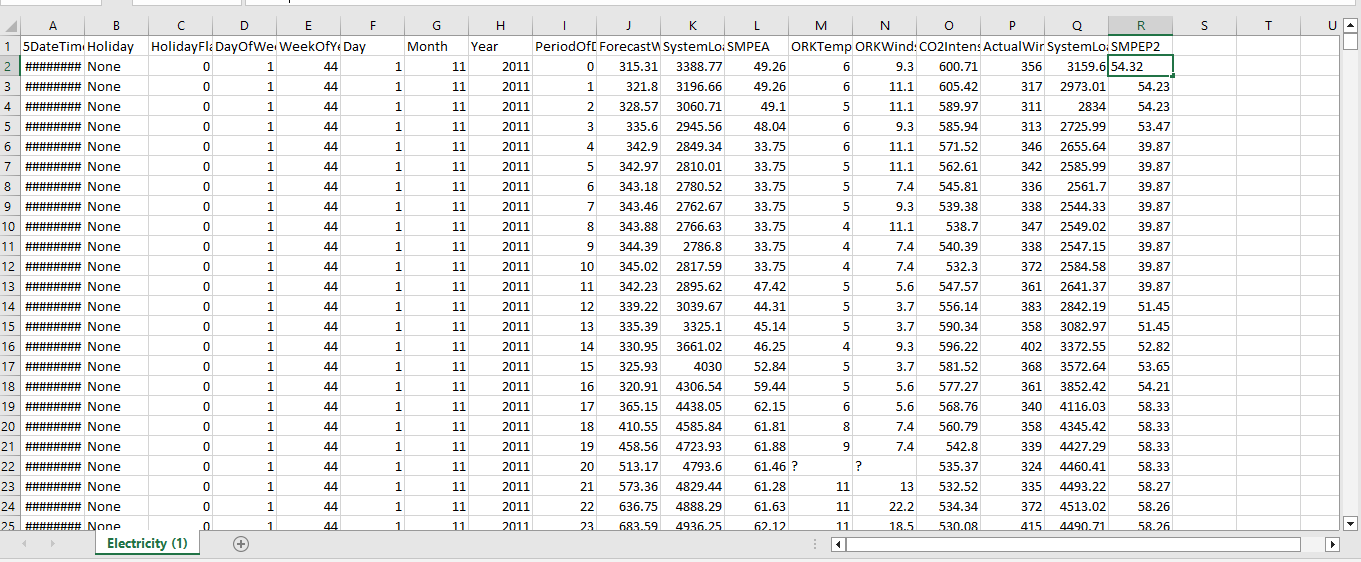
While optimizing data loading and processing is crucial for accurate electricity price prediction, there are several challenges to overcome. Dealing with large and complex datasets requires efficient handling and storage solutions. Additionally, processing speed and scalability are key considerations, as real-time predictions are often required. Addressing these challenges will ensure the successful implementation of machine learning algorithms for electricity price prediction.

**Data Collection:**

First, you need to gather relevant data. You can obtain historical electricity price data from sources like government agencies, utility companies, or open data repositories.

**Data Import:** Once you have your dataset, you need to import it into your data science environment. Common tools include Python with libraries like Pandas for data manipulation and Jupyter notebooks for analysis.

**Data Exploration:** Start by exploring your dataset. Use functions like head(), info(), and describe() to get an initial understanding of the data's structure and content. Brief overview of initial data analysis.Visualizations to understand data distribution and trends.



**DATA SORCE:** [**https://www.kaggle.com/datasets/chakradharmattapalli/electricity-price-prediction**](https://www.kaggle.com/datasets/chakradharmattapalli/electricity-price-prediction)

**DATA LOADING:**

. Dealing with large and complex datasets requires efficient handling and storage solutions. Additionally, processing speed and scalability are key considerations, as real-time predictions are often required. Addressing these challenges will ensure the successful implementation of machine learning algorithms for electricity price prediction.

**INPUT:**

*# Import packages and modules*

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import xgboost as xgb

from xgboost import XGBRegressor, plot\_importance

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error

from sklearn.model\_selection import GridSearchCV

from sklearn.model\_selection import train\_test\_split

from keras import optimizers

from keras.models import Sequential, Model

from keras.layers.convolutional import Conv1D, MaxPooling1D

from keras.layers import Dense, LSTM, RepeatVector, TimeDistributed, Flatten

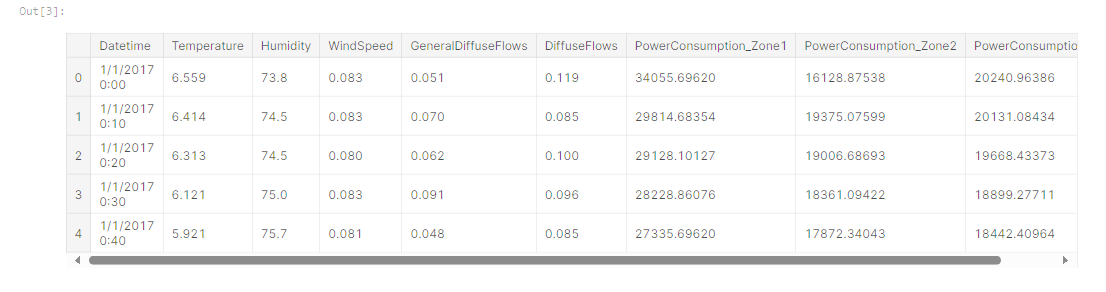
from sklearn.metrics import r2\_score

*# Load Dataset*

df = pd.read\_csv("/kaggle/input/electric-power-consumption/powerconsumption.csv")

df.head() *#Show the first lines of the dataframe*

**OUTPUT:**

****

**DATA PROCESSING :**

**Preprocessing data for electricity price prediction**

To ensure accurate and effective prediction of electricity prices, proper data preprocessing is crucial. This step involves handling missing values, normalizing data, removing outliers, and selecting relevant features. By carefully cleaning and transforming the data before feeding it into the machine learning models, we can improve the accuracy and reliability of our electricity price predictions.

**Optimizing data loading for machine learning models**

When it comes to optimizing data loading for machine learning models, there are several strategies to consider. These include utilizing parallel processing techniques, optimizing data storage formats for efficient retrieval, and leveraging distributed computing platforms.

By implementing these techniques, we can enhance the speed and efficiency of data loading, leading to faster and more accurate electricity price predictions**.**

**Optimizing data processing for accurate predictions**

To ensure accurate predictions in electricity price forecasting, optimizing data processing is crucial. This involves techniques like feature engineering, dimensionality reduction, and data normalization. By applying these methods, we can enhance the predictive power of machine learning models and improve the accuracy of electricity price predictions, enabling more informed decision-making in the energy industry. Addressing missing values and outliers.

Feature engineering: Creating relevant features forprediction.Importance of normalization for consistent model performance.Techniques like Min-Max scaling and Z-score normalization.

Once you have your dataset, you need to import it into your data science environment. Common tools include Python with libraries like Pandas for data manipulation and Jupyter notebooks for analysis.

Handle Missing Data: Identify and address missing values in the dataset using techniques like imputation or removal.

Data Cleaning: Clean the data by removing duplicates, outliers, and irrelevant information.

Feature Engineering:

Create relevant features that can help improve the accuracy of your model. This might include date/time features, weather data, or economic indicators that could influence electricity prices.

def create\_features(df):

*"""*

*Create time series features based on time series index.*

*"""*

df = df.copy()

df['hour'] = df.index.hour

df['minute'] = df.index.minute

df['dayofweek'] = df.index.dayofweek

df['quarter'] = df.index.quarter

df['month'] = df.index.month

df['day'] = df.index.month

df['year'] = df.index.year

df['season'] = df['month'] % 12 // 3 + 1

df['dayofyear'] = df.index.dayofyear

df['dayofmonth'] = df.index.day

df['weekofyear'] = df.index.isocalendar().week

*# Additional features*

df['is\_weekend'] = df['dayofweek'].isin([5, 6]).astype(int)

df['is\_month\_start'] = (df['dayofmonth'] == 1).astype(int)

df['is\_month\_end'] = (df['dayofmonth'] == df.index.days\_in\_month).astype(int)

df['is\_quarter\_start'] = (df['dayofmonth'] == 1) & (df['month'] % 3 == 1).astype(int)

df['is\_quarter\_end'] = (df['dayofmonth'] == df.groupby(['year', 'quarter'])['dayofmonth'].transform('max'))

*# Additional features*

df['is\_working\_day'] = df['dayofweek'].isin([0, 1, 2, 3, 4]).astype(int)

df['is\_business\_hours'] = df['hour'].between(9, 17).astype(int)

df['is\_peak\_hour'] = df['hour'].isin([8, 12, 18]).astype(int)

*# Minute-level features*

df['minute\_of\_day'] = df['hour'] \* 60 + df['minute']

df['minute\_of\_week'] = (df['dayofweek'] \* 24 \* 60) + df['minute\_of\_day']

return df.astype(float)

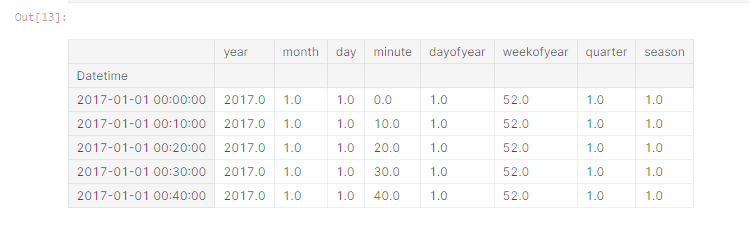
In [12]:

df = df.set\_index('Datetime')

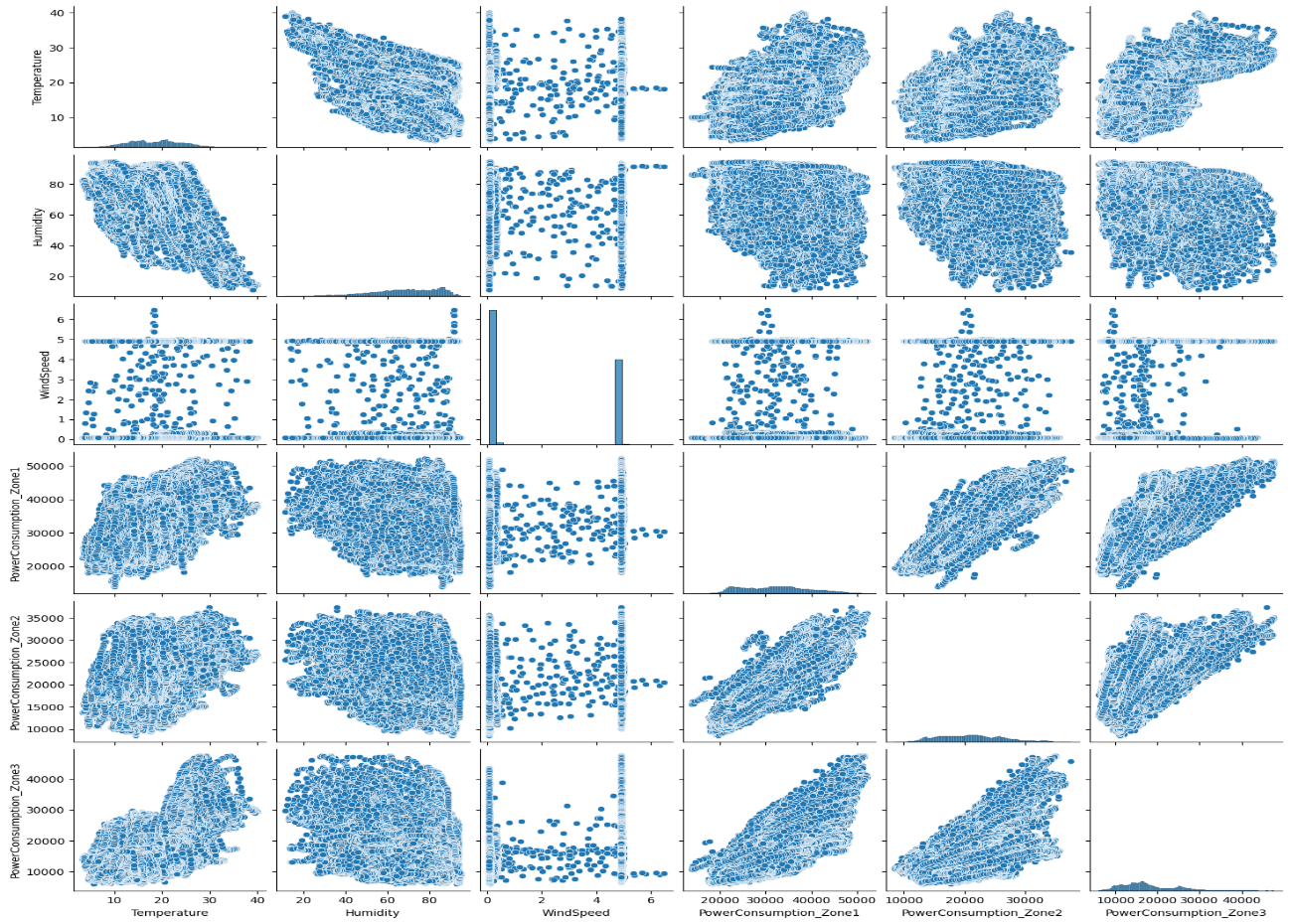
df = create\_features(df)

In [13]:

df[[ 'year', 'month', 'day','minute', 'dayofyear', 'weekofyear', 'quarter', 'season']].head()



Data Visualization: Visualize the dataset to gain insights into its patterns. You can use libraries like Matplotlib and Seaborn for plotting.



**Data Splitting**:

Split your dataset into a training set and a testing set to evaluate your model's performance. Common splits are 70/30 or 80/20 for training/testing, respectively.

**Evaluating the performance of machine learning models:**

To evaluate the performance of machine learning models for electricity price prediction, various metrics need to be considered. These include root mean squared error (RMSE), mean absolute error (MAE), coefficient of determination (R-squared), and precision/recall. By analyzing these metrics, we can determine the effectiveness of our models in accurately forecasting electricity prices and make informed decisions based on their performance.

*# Calculate correlation matrix*

correlation\_matrix = df[['Temperature', 'Humidity', 'WindSpeed', 'PowerConsumption\_Zone1', 'PowerConsumption\_Zone2', 'PowerConsumption\_Zone3']].corr()

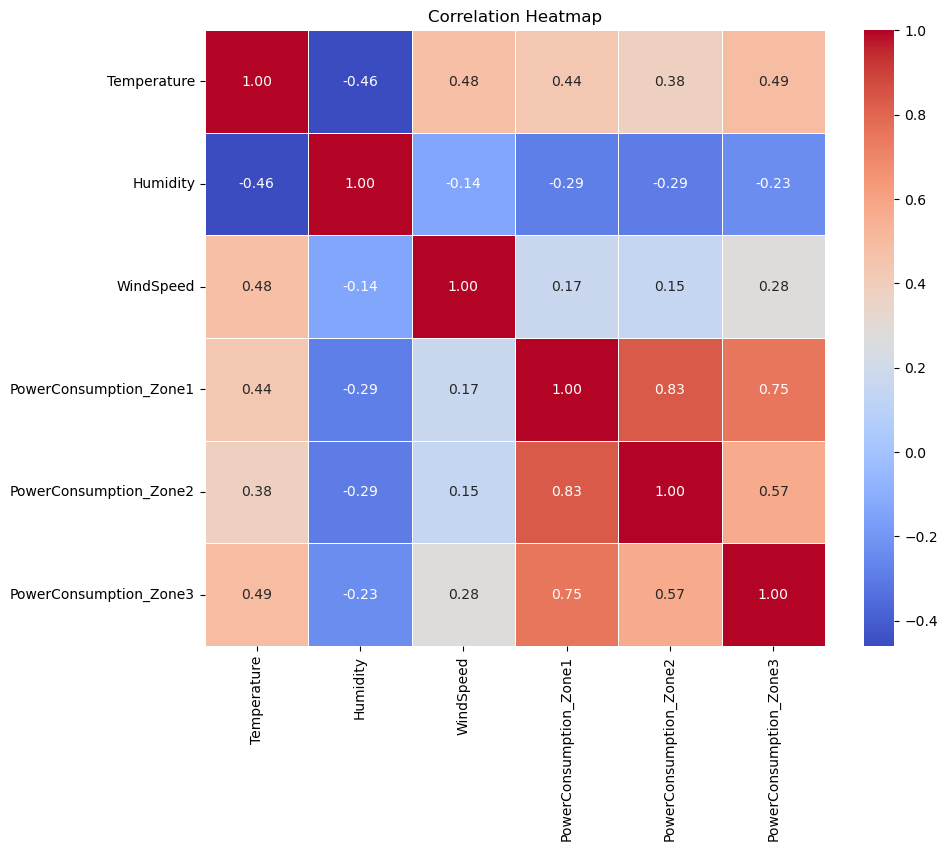
*# Create a heatmap of the correlation matrix*

plt.figure(figsize=(10, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)

plt.title('Correlation Heatmap')

plt.show()



*# Resample the data for more meaningful time series analysis (e.g., daily, weekly)*

daily\_resampled = df.resample('D').mean()

*# Plot daily Power Consumption for each zone*

plt.figure(figsize=(12, 6))

sns.lineplot(data=daily\_resampled[['PowerConsumption\_Zone1', 'PowerConsumption\_Zone2', 'PowerConsumption\_Zone3']])

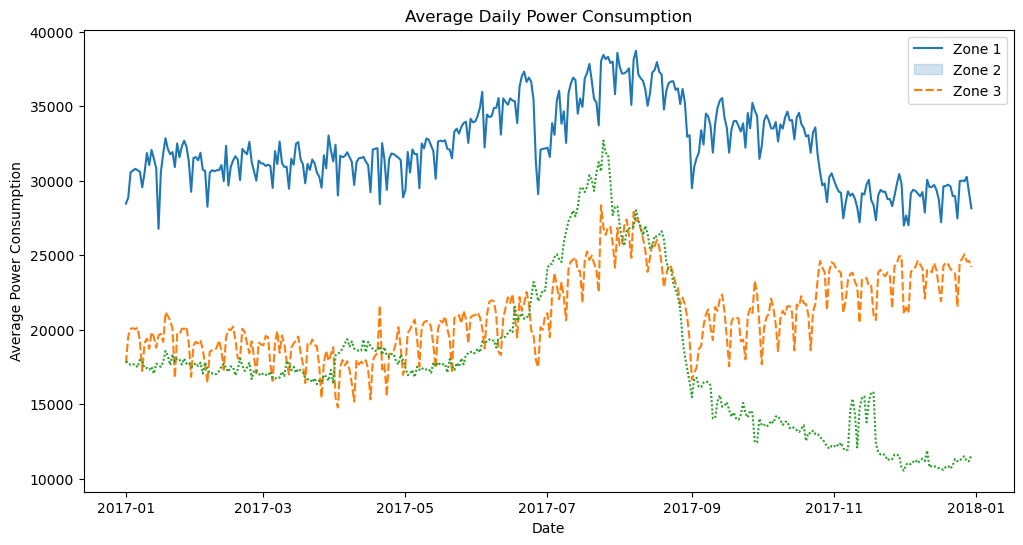
plt.xlabel('Date')

plt.ylabel('Average Power Consumption')

plt.title('Average Daily Power Consumption')

plt.legend(labels=['Zone 1', 'Zone 2', 'Zone 3'])

plt.show()



**MODELING:**

Model Building: Choose a suitable machine learning model for electricity price prediction. Common choices include linear regression, decision trees, random forests, or more advanced models like neural networks.

from sklearn.preprocessing import StandardScaler

*# Separate the input features (X) and target variables (y)*

X = df.drop(['PowerConsumption\_Zone1', 'PowerConsumption\_Zone2', 'PowerConsumption\_Zone3'], axis=1)

y = df[['PowerConsumption\_Zone1', 'PowerConsumption\_Zone2', 'PowerConsumption\_Zone3']]

*# Initialize StandardScaler for y*

scaler\_y = StandardScaler()

*# Fit and transform y*

y\_scaled = scaler\_y.fit\_transform(y)

In [17]:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y\_scaled,

**Model Training**: Train your chosen model on the training dataset. Use appropriate evaluation metrics (e.g., Mean Absolute Error, Root Mean Squared Error) to assess its performance.

**Model Evaluation:** Evaluate the model's performance on the testing dataset to ensure it generalizes well to new data.

**Hyperparameter Tuning**: Optimize your model's hyperparameters to improve its performance. You can use techniques like grid search or random search.

**Model Deployment**: If the model meets your requirements, you can deploy it in a production environment to make real-time predictions.

**Monitoring and Maintenance**: Continuously monitor the model's performance in the production environment and retrain it as needed to keep it up to date.

**Documentation:** Document all your steps, code, and findings for future reference and collaboration.

CONCLUSION:

The proposed methodology effectively forecasts the price and demand for electricity. The choice of the estimation procedure used for both deterministic and stochastic components has a significant effect on the forecasting results1.

The proposed method outperforms other methods in prediction accuracy and spike-capturing ability, with EEMD reducing the mean absolute percentage error (MAPE) by 53%, 54%, and 60%, respectively. In the three forecast periods, the average MAPE and are 0.097 and 0.92, respectively2.

The prediction accuracy of the deep belief network model is higher, and the use of the deep belief network can provide an effective method for China’s electricity sales companies to predict electricity prices3.

The proposed method and model can provide a valuable tool for data processing and forecasting. Compared with DE-SVM without data processing, the forecasting accuracy is improved from 81.68% to 91.44%, and the time-cost is decreased from 35,074 s to 1,809 s4.