

ELECTRICITY PRICES PREDICTION

(GROUP 2-PHASE 4)

DEVELOPMENT PART 2

**TEAM MEMBERS:**

*1. PRAVEENA.E*

*2. PAVITHRA.S*

*3. NISHA.J*

Building an electricity price prediction model involves several steps, including data

preprocessing, feature engineering, model training, and evaluation. Below each of these steps in more detail:

**1. Data Preprocessing:**

- Load the dataset: You can use the Pandas library to load the dataset from the provided

link.

```python

import pandas as pd

data = pd.read\_csv(&quot;your\_dataset\_path.csv&quot;)

```

- Explore the data: Get an understanding of the data by examining its structure, checking

for missing values, and performing basic statistics and data visualization.

```python

data.info()

data.describe()

data.head()

```

- Handle missing values: Depending on the dataset, you may need to deal with missing

values. You can choose to impute missing data or drop rows/columns with too many missing

values.

- Convert date and time columns: If your dataset contains date and time information,

consider converting them into a datetime format. This allows you to extract features like

day of the week, month, hour, etc., which can be useful for modeling.

**2. Feature Engineering:**

- Create new features: Based on domain knowledge, create new features that could

potentially be predictive of electricity prices. For example, you might want to create lag

features, rolling statistics, or one-hot encode categorical variables.

- Feature selection: Not all features are equally relevant. Use techniques like correlation

analysis or feature importance from machine learning models to select the most informative

features.

**3. Model Training:**

- Split the data: Split your dataset into training and testing sets. You can use the

`train\_test\_split` function from Scikit-learn.

```python

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

X = data.drop(&#39;target\_column\_name&#39;, axis=1) # Features

y = data[&#39;target\_column\_name&#39;] # Target variable

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

```

- Choose a model: Select a machine learning model suitable for regression tasks. Common

choices include Linear Regression, Decision Trees, Random Forests, Gradient Boosting, or

Neural Networks.

- Train the model: Fit the selected model to the training data.

```python

from sklearn.linear\_model import LinearRegression # Replace with your chosen model

model = LinearRegression() # Replace with your chosen model

model.fit(X\_train, y\_train)

```

**4. Evaluation:**

- Predict electricity prices: Use the trained model to make predictions on the test dataset.

```python

y\_pred = model.predict(X\_test)

```

- Evaluate the model: Use appropriate regression metrics to assess the model&#39;s

performance. Common metrics include Mean Absolute Error (MAE), Mean Squared Error

(MSE), Root Mean Squared Error (RMSE), and R-squared (R2) score.

```python

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

mae = mean\_absolute\_error(y\_test, y\_pred)

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)

r2 = r2\_score(y\_test, y\_pred)

```

- Visualize results: Consider visualizing the actual vs. predicted values to get a better

understanding of the model&#39;s performance.

**5. Fine-Tuning and Deployment (Optional):**

- Depending on the results, you may want to fine-tune hyperparameters or try different

models to improve performance.

- If the model performs well, you can deploy it for real-world predictions.

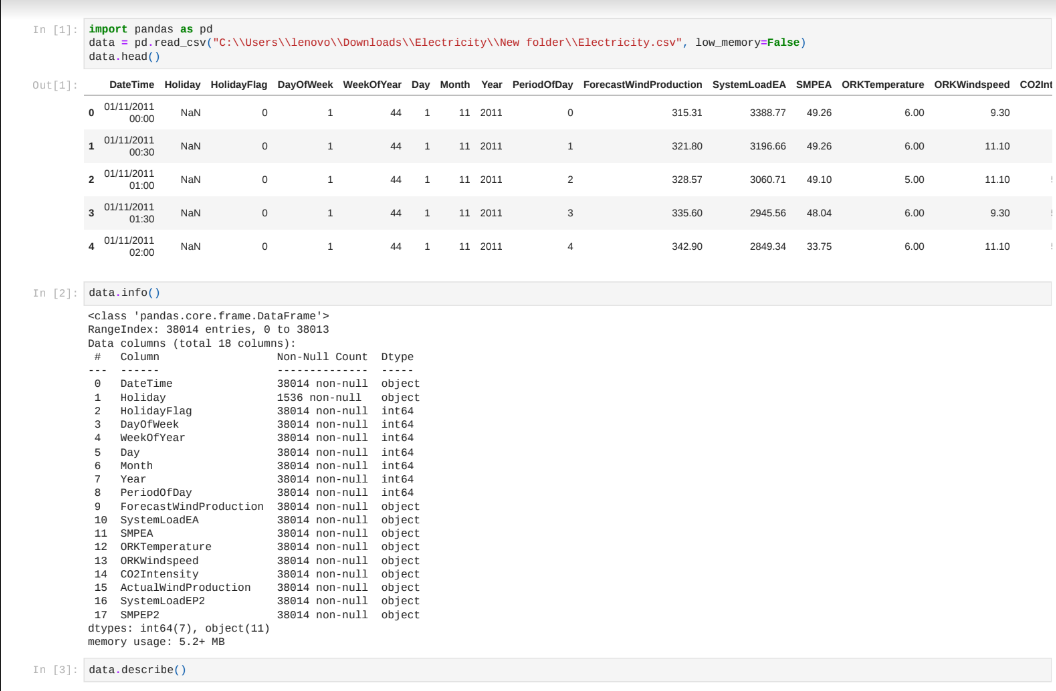
Remember that building an effective prediction model often involves iterative steps, and

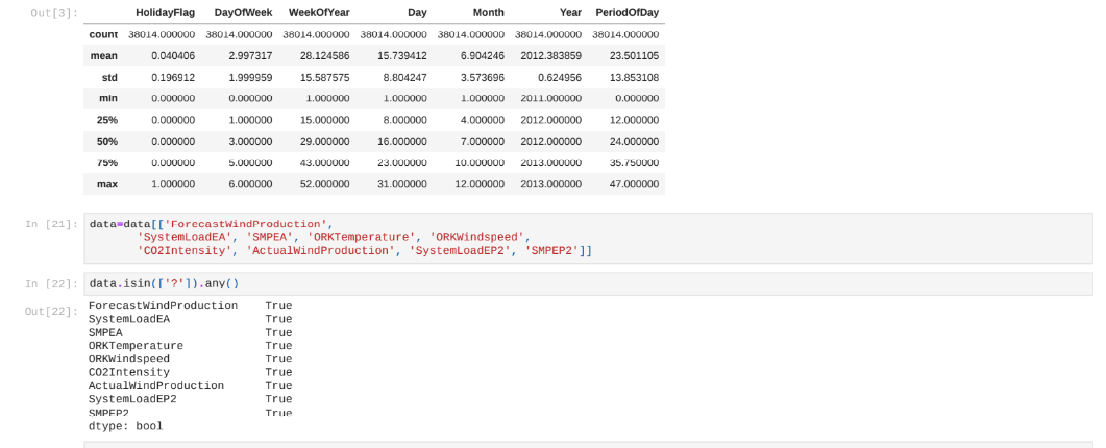
you may need to try different models and feature engineering techniques to achieve the

best results. Additionally, you can use libraries like Scikit-learn and TensorFlow/Keras (for

deep learning) to streamline the modeling process.

PROGRAM:













**THANK YOU**

SUBMITTED BY-

PAVITHRA.S