KINGSTON

ENGINEERING COLLEGE

ELECTRICITY PRICES PREDICTION

(GROUP 2-PHASE 3)

Development part-1

SUBMITTED BY-

S.PAVITHRA

Download the Dataset:

Go to the Kaggle dataset link you provided and download the dataset in a format such as CSV.

Install Required Libraries:

Ensure that you have the necessary libraries installed, such as Pandas, NumPy, and scikit-learn. You can install them using pip:

```
pip install pandas numpy scikit-learn
```

Load the Dataset:

Use Pandas to load the dataset into a DataFrame:

```
import pandas as pd

# Replace 'your_dataset.csv' with the actual file path of the downloaded dataset

df = pd.read_csv('your_dataset.csv')
```

Explore the Data:

It's essential to understand the dataset before preprocessing. You can check the first few rows of the dataset, data types, and summary statistics using functions like `head()`, `info()`, and ‘describe()’

```
print(df.head())

print(df.info())

print(df.describe())
```

Data Preprocessing:

Depending on the dataset and the specific requirements of your electricity price prediction model, you may need to perform various preprocessing tasks. Common preprocessing steps include:

- Handling missing values (e.g., using `fillna()` or dropping rows/columns).
- Handling categorical data (e.g., encoding with one-hot encoding or label encoding).
- Scaling or normalizing numerical features.

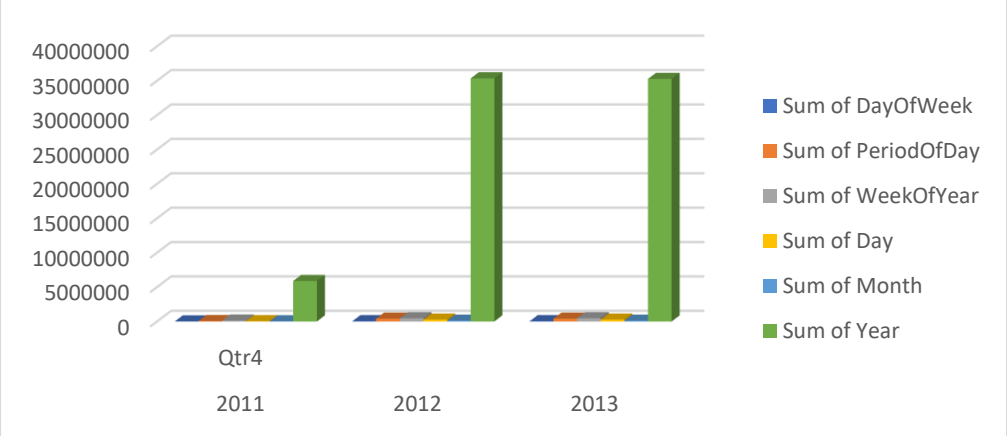
- Splitting the dataset into features (X) and target

Model Building:

After preprocessing the data, you can proceed to build your electricity price prediction model using machine learning or deep learning techniques, depending on your project's requirements.

Remember to adjust the preprocessing steps based on the characteristics of your dataset and the goals of your prediction model. If your dataset has specific characteristics or challenges, further preprocessing steps may be needed.

Sum of DayOfWeek	Sum of PeriodOfDay	Sum of WeekOfYear	Sum of Day	Sum of Month	Sum of Year
8784	68808	140544	46128	33696	5888208
52692	412843	465528	276766	114426	35342792
52464	411720	463056	275424	114336	35267760
113940	893371	1069128	598318	262458	76498760



Program:

```
"C:\Users\prath\Downloads\Electricity.csv"
```

Python

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
```

Python

```
dataset_path = "C:\\Users\\prath\\Downloads\\Electricity.csv"
data = pd.read_csv((dataset_path),low_memory=False)
data.head()
```

Python

```
data=data[['ForecastWindProduction',
       'SystemLoadEA', 'SMPEA', 'ORKTemperature', 'ORKWindspeed',
       'CO2Intensity', 'ActualWindProduction', 'SystemLoadEP2', 'SMPEP2']]
```

Python

```
data.isin(['?']).any()
```

Python

```
ForecastWindProduction    True
SystemLoadEA              True
SMPEA                     True
ORKTemperature            True
ORKWindspeed              True
CO2Intensity              True
ActualWindProduction      True
SystemLoadEP2             True
SMPEP2                    True
dtype: bool
```

```
for col in data.columns:
    data.drop(data.index[data[col] == '?'], inplace=True)
```

Python

```
data=data.apply(pd.to_numeric)
data=data.reset_index()
data.drop('index', axis=1, inplace=True)
```

Python

```
data.info()
```

Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37682 entries, 0 to 37681
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ForecastWindProduction  37682 non-null  float64
 1   SystemLoadEA            37682 non-null  float64
 2   SMPEA                   37682 non-null  float64
 3   ORKTemperature          37682 non-null  float64
 4   ORKWindspeed            37682 non-null  float64
 5   CO2Intensity            37682 non-null  float64
 6   ActualWindProduction    37682 non-null  float64
 7   SystemLoadEP2           37682 non-null  float64
 8   SMPEP2                  37682 non-null  float64
dtypes: float64(9)
memory usage: 2.6 MB
```

```
data.corrwith(data['SMPEP2']).abs().sort_values(ascending=False)
```

Python

```
SMPEP2                  1.000000
SMPEA                   0.618158
SystemLoadEP2           0.517081
SystemLoadEA            0.491096
ActualWindProduction    0.083434
ForecastWindProduction  0.079639
ORKWindspeed            0.035436
CO2Intensity            0.035055
ORKTemperature          0.009087
dtype: float64
```

```
X=data.drop('SMPEP2', axis=1)
y=data['SMPEP2']
```

Python

```
x_train, x_test, y_train, y_test=train_test_split(X,y, test_size=0.2, random_state=42)
```

Python

```
linear_model=LinearRegression()
linear_model.fit(x_train, y_train)
linear_predict=linear_model.predict(x_test)
np.sqrt(mean_squared_error(y_test, linear_predict))
```

Python

```
27.862965246485324
```

```
forest_model=RandomForestRegressor()
forest_model.fit(x_train, y_train)
forest_predict=forest_model.predict(x_test)
print(np.sqrt(mean_squared_error(y_test, forest_predict)))
```

Python

```
25.198701853469586
```

```
tree_model=DecisionTreeRegressor(max_depth=50)
tree_model.fit(x_train, y_train)
tree_predict=tree_model.predict(x_test)
print(np.sqrt(mean_squared_error(y_test, tree_predict)))
```

Python

```
33.76792802500666
```

```
knn_model=KNeighborsRegressor()
knn_model.fit(x_train, y_train)
knn_predict=knn_model.predict(x_test)
print(np.sqrt(mean_squared_error(y_test, knn_predict)))
```

Python

```
28.533256274003907
```

```
#Let's see some sample prediction and difference between label and prediction
some_data=x_test.iloc[50:60]
some_data_label=y_test.iloc[50:60]
some_predict=forest_model.predict(some_data)
pd.DataFrame({'Predict':some_predict,'Label':some_data_label})
```

Python

Thank you