Phase 5

712221205032 / Santhosh Kumar P

712221205018 / Mukesh Kumar Pandian M

712221205021 / Naveen P

712221205038 / Srikaran S

Electricity Price Prediction using Python:

I will start the task of electricity price prediction by importing the necessary Python libraries and the dataset that we need for this task:

## Data Preprocessing

Import pandas as pd

Import numpy as np

Data=pd.read_csv(https://github.com/Santhoshpain/Datasciencephase1/blob/a16cf068241c7eb7b9b24f8e05523851cbe6adde/Electricity.csv)

Print(data.head())

```
DateTime Holiday  ... SystemLoadEP2  SMPEP2
0  01/11/2011 00:00    None  ...       3159.60   54.32
1  01/11/2011 00:30    None  ...       2973.01   54.23
2  01/11/2011 01:00    None  ...       2834.00   54.23
3  01/11/2011 01:30    None  ...       2725.99   53.47
4  01/11/2011 02:00    None  ...       2655.64   39.87

[5 rows x 18 columns]
```

Let's have a look at all the columns of this dataset:

Data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38014 entries, 0 to 38013
Data columns (total 18 columns):
 #   Column              Non-Null Count  Dtype
```

```
---  ------                --------------  -----
 0   DateTime              38014 non-null  object
 1   Holiday               38014 non-null  object
 2   HolidayFlag           38014 non-null  int64
 3   DayOfWeek             38014 non-null  int64
 4   WeekOfYear            38014 non-null  int64
 5   Day                   38014 non-null  int64
 6   Month                 38014 non-null  int64
 7   Year                  38014 non-null  int64
 8   PeriodOfDay           38014 non-null  int64
 9   ForecastWindProduction  38014 non-null  object
 10  SystemLoadEA          38014 non-null  object
 11  SMPEA                 38014 non-null  object
 12  ORKTemperature        38014 non-null  object
 13  ORKWindspeed          38014 non-null  object
 14  CO2Intensity          38014 non-null  object
 15  ActualWindProduction  38014 non-null  object
 16  SystemLoadEP2         38014 non-null  object
 17  SMPEP2                38014 non-null  object
Dtypes: int64(7), object(11)
Memory usage: 5.2+ MB
```

I can see that so many features with numerical values are string values in the dataset and not integers or float values. So before moving further, we have to convert these string values to float values:

```
Data["ForecastWindProduction"] = pd.to_numeric(data["ForecastWindProduction"], errors= 'coerce')

Data["SystemLoadEA"] = pd.to_numeric(data["SystemLoadEA"], errors= 'coerce')

Data["SMPEA"] = pd.to_numeric(data["SMPEA"], errors= 'coerce')

Data["ORKTemperature"] = pd.to_numeric(data["ORKTemperature"], errors= 'coerce')
```

Data[“ORKWindspeed”] = pd.to_numeric(data[“ORKWindspeed”], errors= ‘coerce’)

Data[“CO2Intensity”] = pd.to_numeric(data[“CO2Intensity”], errors= ‘coerce’)

Data[“ActualWindProduction”] = pd.to_numeric(data[“ActualWindProduction”], errors= ‘coerce’)

Data[“SystemLoadEP2”] = pd.to_numeric(data[“SystemLoadEP2”], errors= ‘coerce’)

Data[“SMPEP2”] = pd.to_numeric(data[“SMPEP2”], errors= ‘coerce’)

Now let’s have a look at whether this dataset contains any null values or not:

Data.isnull().sum()

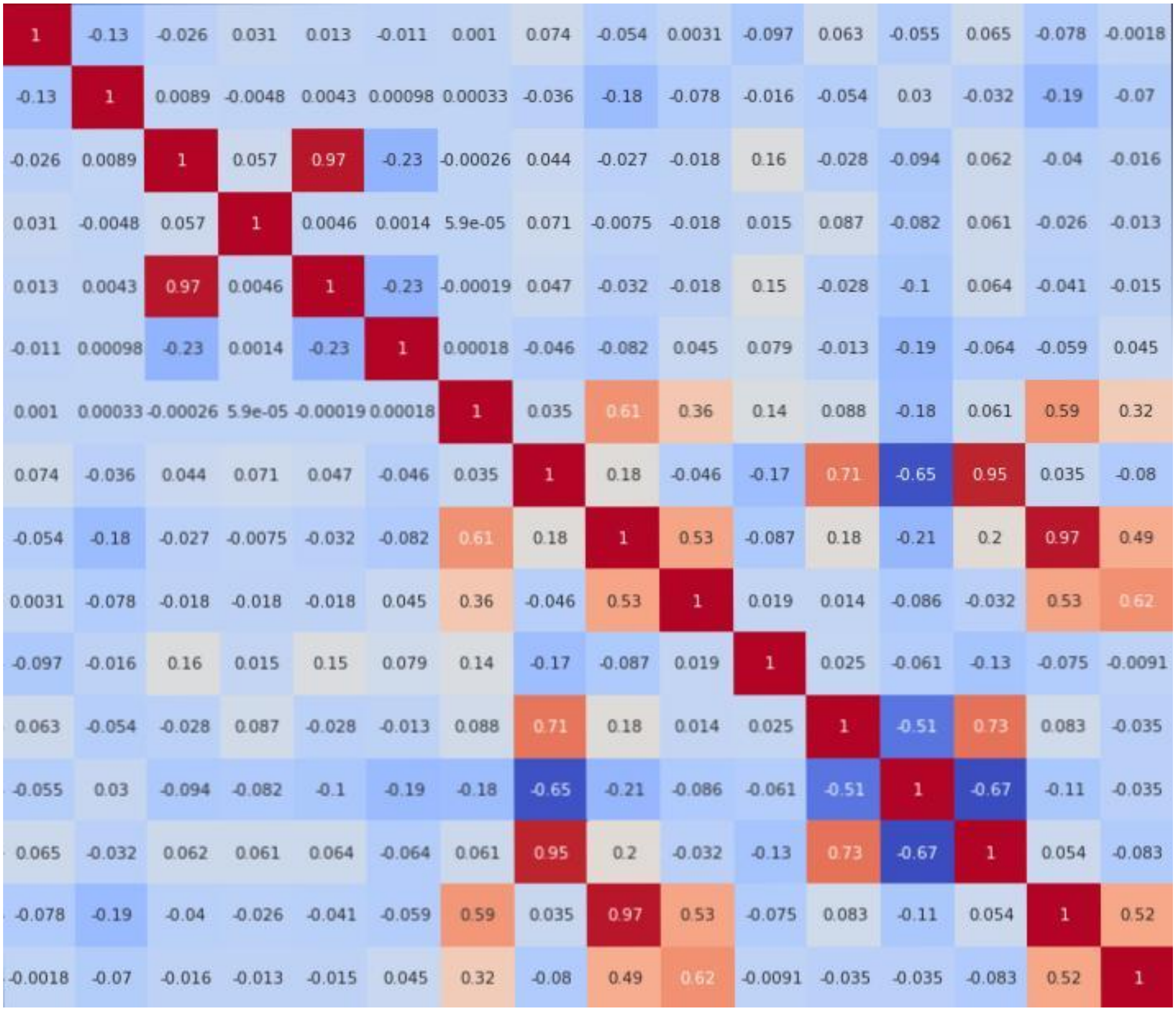
```
DateTime                 0
Holiday                  0
HolidayFlag              0
DayOfWeek                0
WeekOfYear               0
Day                      0
Month                    0
Year                     0
PeriodOfDay              0
ForecastWindProduction   5
SystemLoadEA             2
SMPEA                    2
ORKTemperature         295
ORKWindspeed           299
CO2Intensity             7
ActualWindProduction     5
SystemLoadEP2            2
SMPEP2                   2
Dtype: int64
```

So there are some columns with null values, I will drop all these rows containing null values from the dataset:

Data = data.dropna()

Now let’s have a look at the correlation between all the columns in the dataset:

```
Import seaborn as sns

Import matplotlib.pyplot as plt

Correlations = data.corr(method=’pearson’)

Plt.figure(figsize=(16, 12))

Sns.heatmap(correlations, cmap=”coolwarm”, annot=True)

Plt.show()
```




Electricity Price Prediction Model:

Now let's move to the task of training an electricity price prediction model. Here I will first add all the important features to x and the target column to y, and then I GT will split the data into training and test sets:

Model Training Process

```
X = data[["Day", "Month", "ForecastWindProduction", "SystemLoadEA",
    "SMPEA", "ORKTemperature", "ORKWindspeed", "CO2Intensity",
    "ActualWindProduction", "SystemLoadEP2"]]
```

Y = data[“SMPEP2”]

From sklearn.model_selection import train_test_split

Xtrain, xtest, ytrain, ytest = train_test_split(x, y,

Test_size=0.2,

Random_state=42)

As this is the problem of regression, so here I will choose the Random Forest regression algorithm to train the electricity price prediction model:

From sklearn.ensemble import RandomForestRegressor

Model = RandomForestRegressor()

Model.fit(xtrain, ytrain)

RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion=’mse’,

Max_depth=None, max_features=’auto’, max_leaf_nodes=None,

Max_samples=None, min_impurity_decrease=0.0,

Min_impurity_split=None, min_samples_leaf=1,

Min_samples_split=2, min_weight_fraction_leaf=0.0,

N_estimators=100, n_jobs=None, oob_score=False,

Random_state=None, verbose=0, warm_start=False

Now let’s input all the values of the necessary features that we used to train the model and have a look at the price of the electricity predicted by the model:

#features = [[“Day”, “Month”, “ForecastWindProduction”, “SystemLoadEA”, “SMPEA”, “ORKTemperature”, “ORKWindspeed”, “CO2Intensity”, “ActualWindProduction”, “SystemLoadEP2”]]

Features = np.array([[10, 12, 54.10, 4241.05, 49.56, 9.0, 14.8, 491.32, 54.0, 4426.84]])

Model.predict(features)

Array([65.1696])

So this is how you can train a machine learning model to predict the prices of electricity.