Electricity Price Prediction with Machine Learning

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.

Problem definition

Suppose that your business relies on computing services where the power consumed by your machines varies throughout the day. You do not know the actual cost of the electricity consumed by the machines throughout the day, but the organization has provided you with historical data of the price of the electricity consumed by the machines. Below is the information of the data we have for the task of forecasting electricity prices:

DateTime: Date and time of the record
Holiday: contains the name of the holiday if the day is a national holiday
HolidayFlag: contains 1 if it's a bank holiday otherwise 0
DayOfWeek: contains values between 0-6 where 0 is Monday
WeekOfYear: week of the year
Day: Day of the date
Month: Month of the date
Year: Year of the date
PeriodOfDay: half-hour period of the day
ForcastWindProduction: forecasted wind production
SystemLoadEA forecasted national load
SMPEA: forecasted price
ORKTemperature: actual temperature measured
ORKWindspeed: actual windspeed measured
CO2Intensity: actual C02 intensity for the electricity produced
ActualWindProduction: actual wind energy production
SystemLoadEP2: actual national system load
SMPEP2: the actual price of the electricity consumed (labels or values to be predicted)

So your task here is to use this data to train a machine learning model to predict the price of electricity consumed by the machines. In the section below, I will take you through the task of electricity price prediction with machine learning using Python.

Designing and thinking

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:

```
import pandas as pd
import numpy as np
data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/Website-data/master/electricit
y.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')
view rawelectricity1.py hosted with ❤ by GitHub
Now let’s have a look at whether this dataset contains any null values or not:

1
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:

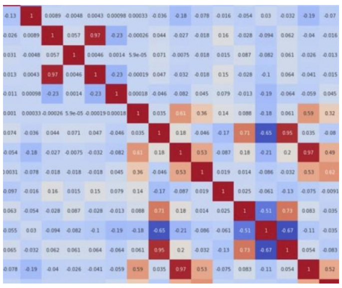
```
data = data.dropna()
```

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()
```

view rawelectricity2.py hosted with ❤ by GitHub

Electricity Price Prediction: correlation



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 will split the data into training and test sets:

```
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)
```

view rawelectricity3.py hosted with ❤ by GitHub

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.

Conclusion

Predicting the price of electricity helps a lot of companies to understand how much electricity expenses they have to pay every year.