## horizontal line



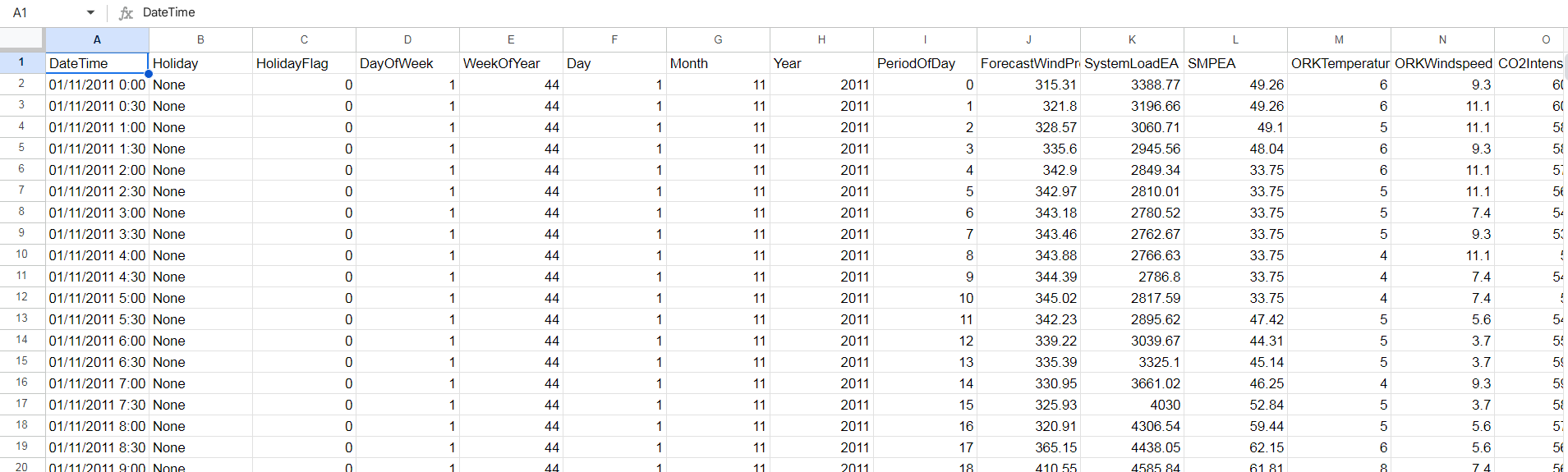
Electricity Price Prediction

**─**

# Overview

Suppose the power consumed by the 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 historical data of the price of the electricity consumed by the machines. Below is the information of the [**data**](https://raw.githubusercontent.com/amankharwal/Website-data/master/electricity.csv) 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 forecasting national load
* SMPEA: forecasted price
* ORKTemperature: actual temperature measured
* ORKWindspeed: actual wind speed measured
* CO2 Intensity: 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)



# Source Code

import numpy as np

import pandas as pd

import os

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

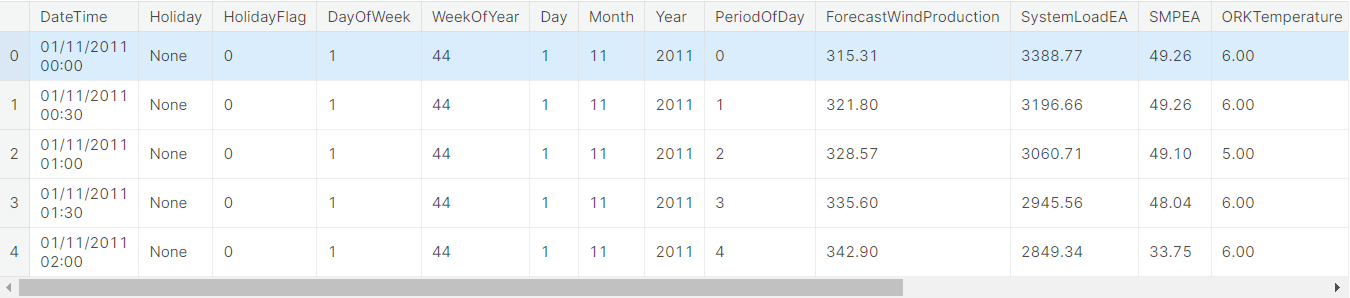
from sklearn.tree import DecisionTreeRegressor

from sklearn.linear\_model import LinearRegression

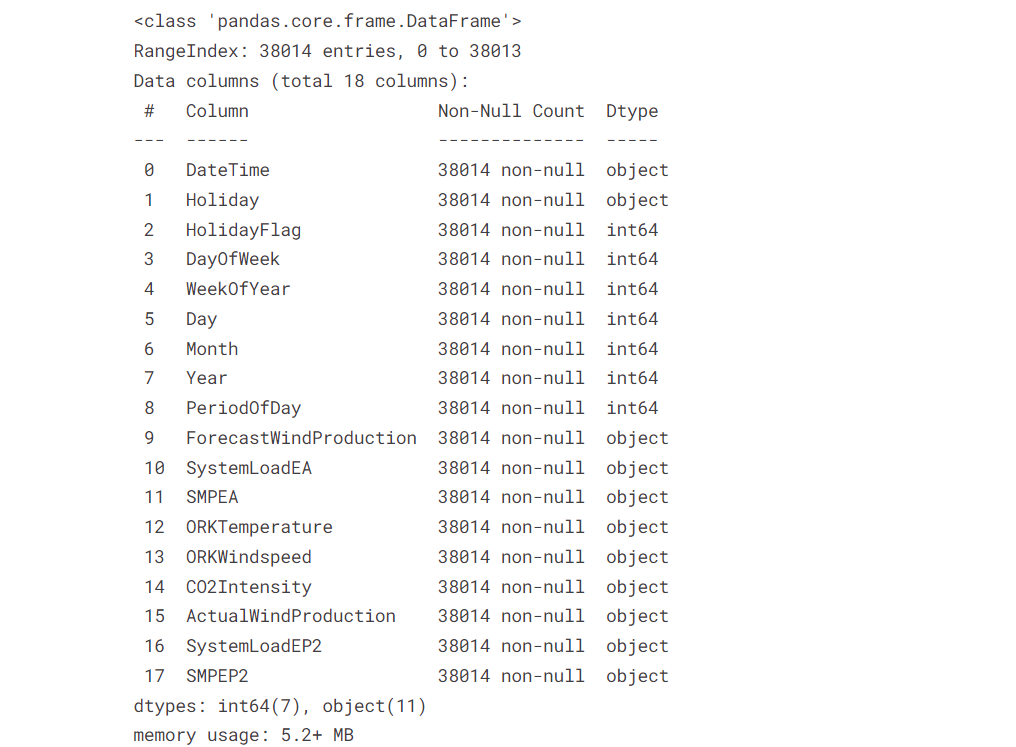
from sklearn.neighbors import KNeighborsRegressor

df=pd.read\_csv("https://github.com/K-2804/Kiran-my-repo/electricity\_price\_data.csv", low\_memory=False)

df.head()



df.info()



data=df[['ForecastWindProduction',

'SystemLoadEA', 'SMPEA', 'ORKTemperature', 'ORKWindspeed',

'CO2Intensity', 'ActualWindProduction', 'SystemLoadEP2', 'SMPEP2']]

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

for col **in** data.columns:

data.drop(data.index[data[col] == '?'], inplace=True)

data=data.apply(pd.to\_numeric)

data=data.reset\_index()

data.drop('index', axis=1, inplace=True)

data.corrwith(data['SMPEP2']).abs().sort\_values(ascending=False)

X=data.drop('SMPEP2', axis=1)

y=data['SMPEP2']

#To Machine Learning

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

#LinearRegression

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))

#RandomForestRegressor

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)))

#DecisionTreeRegressor

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)))

#KNeighborsRegressor

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)))

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})