# Innovation for  Electricity price prediction

## Introduction 0f  Electricity price prediction:

Electricity price prediction is the process of using various techniques, such as statistical analysis, machine learning, and data modeling, to forecast the future cost of electricity. This prediction is vital for consumers, businesses, and energy companies to make informed decisions regarding energy consumption, cost management, and resource allocation. It considers factors like historical data, supply and demand dynamics, weather conditions, renewable energy integration, and consumer behavior to provide accurate estimates of future electricity prices. Accurate price predictions support cost savings, efficient energy management, and the integration of renewable energy sources into the grid.

## About Dataset:

**Dataset link:**https://www.kaggle.com/datasets/chakradharmattapalli/electricity-price-prediction

This is the link about the  Electricity price prediction ,this link is got form www.kaggle.com/data.

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.

This dataset contains 38015 rows and 18 columns. By which It is useful for electricity price prediction.

## Details about columns:

- Date & Time: Date and time of the record
- Holiday: contains the name of the holiday if the day is a national holiday
- Holiday  Flag: contains 1 if it's a bank holiday otherwise 0
- Day Of Week: contains values between 0-6 where 0 is Monday
- Week Of Year: week of the year
- Day: Day of the date
- Month: Month of the date
- Year: Year of the date
- Period Of Day: half-hour period of the day
- For cast Wind Production: forecasted wind production
- System Load EA :forecasted national load
- SMPEA: forecasted price
- ORK Temperature: actual temperature measured
- ORK Windspeed: actual windspeed measured
- CO2Intensity: actual C02 intensity for the electricity produced

- Actual Wind Production: actual wind energy production
- SystemLoadEP2: actual national system load
- SMPEP2: the actual price of the electricity consumed (labels or values to be predicted)

### Details of libraries:

**Reducing electricity costs typically involves optimizing energy consumption, improving energy efficiency, and sometimes integrating renewable energy sources. Python offers a wide range of libraries and tools that can be used in various aspects of electricity price reduction. Here are some key libraries and tools:**

**Pandas: Pandas is essential for data manipulation and analysis. You can use it to preprocess and analyze electricity consumption data, perform data-driven decisions, and forecast energy usage.**

**How to install:**

**Step 1: Open your command prompt or terminal.**

**Step 2: Check if pip is installed:**

**Step 3: pip install pandas**

**Step 4: Wait for the installation to finish.**

**Step 5: Verify the installation by running a Python script that includes:**

```
import pandas as pd

print(pd.version)
```

**If you see the version number, pandas is successfully installed.**

**Num Py :  NumPy is crucial for numerical computing. It can be used for mathematical calculations related to electricity usage and cost reduction strategies.**

**How to install:**

**Step 1: Open your command prompt or terminal.**

**Step 2: Check if pip is installed:**

**Step 3: pip install NumPy**

**Step 4: Wait for the installation to finish.**

**Step 5: Verify the installation by running a Python script that includes:**

```
import numpy as np
```

**print(np.__version__)**

**If you see the version number, pandas is successfully installed.**

**Matplotlib and Seaborn: These libraries are excellent for data visualization. They can help you present insights and trends in a clear and understandable way.**

**How to install:**

**Step 1: Open your command prompt or terminal.**

**Step 2: Check if pip is installed:**

**Step 3: pip install matplotlib**

**Step 4: Wait for the installation to finish.**

**Step 5: Verify the installation by running a Python script that includes:**

**import matplotlib**

**print(matplotlib.__version__)**

**If you see the version number, pandas is successfully installed.**

**Sklearn: Scikit -learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.**

**How to install:**

**Step 1: Open your command prompt or terminal.**

**Step 2: Check if pip is installed:**

**Step 3: pip install scikit-learn**

**Step 4: Wait for the installation to finish.**

**Step 5: Verify the installation by running a Python script that includes:**

**import sklearn**

**print(sklearn.__version__)**

**If you see the version number, pandas is successfully installed.**

**How to train and test dataset:**

**The train-test split is used to estimate the performance of machine learning algorithms that are applicable for prediction-based Algorithms/Applications. This method is a fast and easy procedure to perform such that we can compare our own machine learning model results to machine results.**

**By default, the Test set is split into 30 % of actual data and the training set is split into 70% of the actual data.**
**We need to split a dataset into train and test sets to evaluate how well our machine learning model performs. The train set is used to fit the model, and the statistics of the train set are known. The second set is called the test data set, this set is solely used for predictions.**

**Syntax:**
**train_test_split(*arrays, test_size=None, train_size=None,**
**random_state=None, shuffle=True, stratify=None)**

**code:**

```
# import modules
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

# read the dataset
df = pd.read_csv("Electricity price.csv")

# get the locations
X = df.iloc[:, :-1]
y = df.iloc[:, -1]

# split the dataset
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.05, random_state=0)
```

## Metrics used for the accuracy check for electricity price prediction:

**Evaluation metrics are tied to machine learning tasks, There are different metrics for the tasks classification and regression. Some metrics, like precision-recall, are useful for multiple tasks. Classification and regression are examples of supervised learning, which constitutes a machine learning applications. Using different metrics for performance evaluation, we should to improve our model's overall predictive power before we roll it out for production on unseen Without doing a proper evaluation of the Machine Learning model by using different metrics, and only depending on accuracy, can lead to a problem when the respective model is deployed on unseen data and may end in poor predictions.**
**Classification Metrics in Machine Learning**

**Classification is about predicting the class labels given input data. In binary classification, there are only two possible output classes(i.e., Dichotomy). In multiclass classification, more than two possible classes can be present. I'll focus only on binary classification**
**Accuracy**
**Accuracy simply measures how often the classifier correctly predicts. We**

**can define accuracy as the ratio of the number of correct predictions and the total number of predictions.**

**The four commonly used metrics for evaluating classifier performance are:**

**1. Accuracy: The proportion of correct predictions out of the total predictions.**

**2. Precision: The proportion of true positive predictions out of the total positive predictions (precision = true positives / (true positives + false positives)).**

**3. Recall (Sensitivity or True Positive Rate): The proportion of true positive predictions out of the total actual positive instances (recall = true positives / (true positives + false negatives)).**

**4. F1 Score: The harmonic mean of precision and recall, providing a balance between the two metrics (F1 score = 2 * ((precision * recall) / (precision + recall))).**