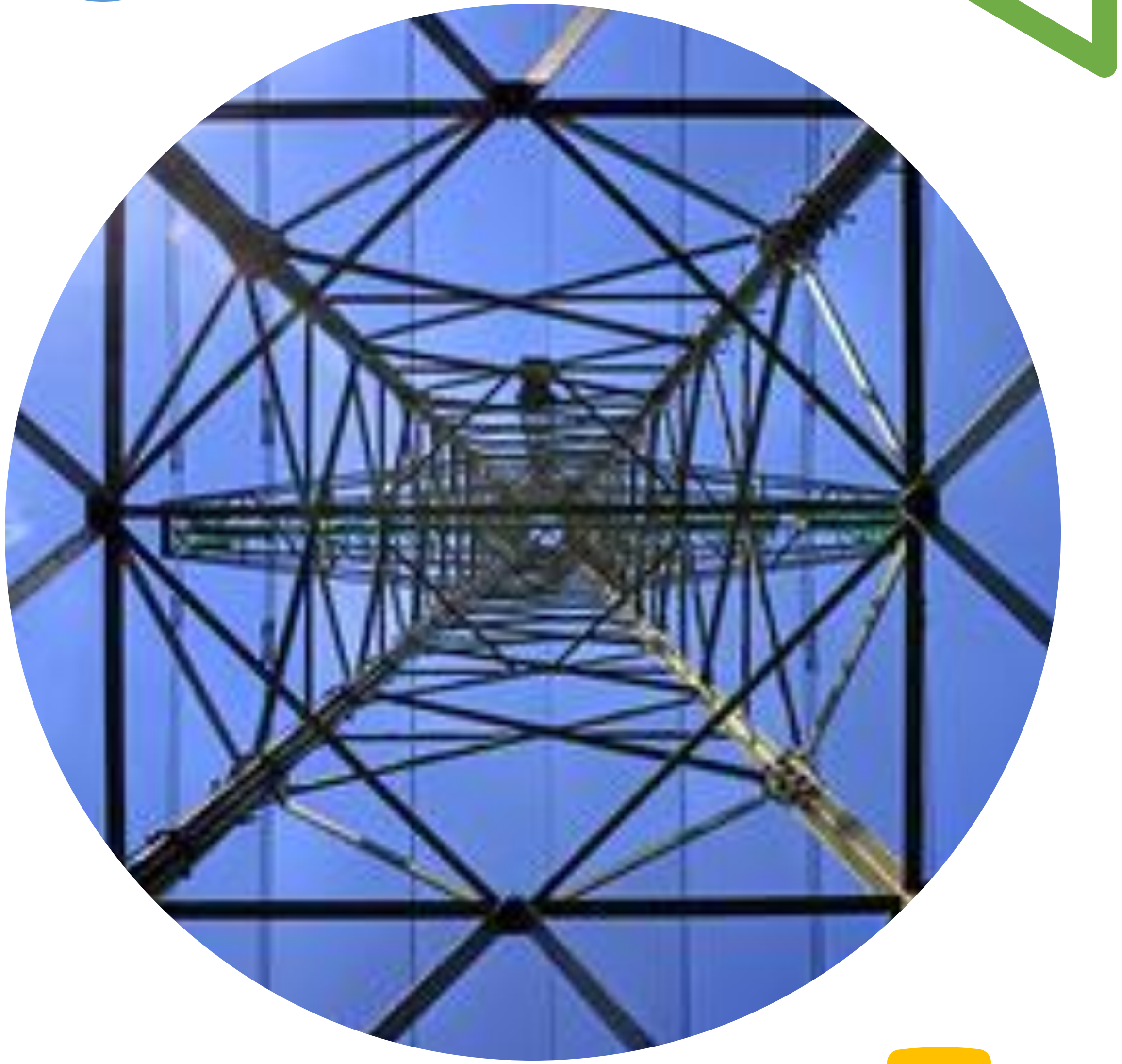
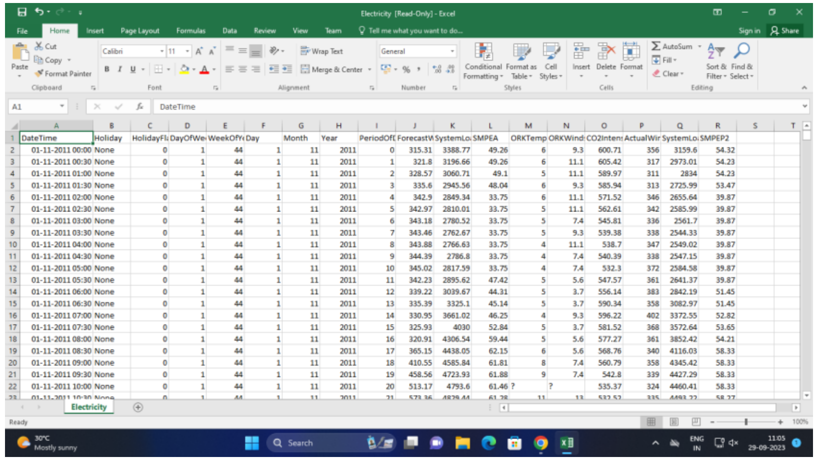


Project Title: Electricity Prices Prediction

Problem Definition:

The problem you've described is a classic example of time series forecasting, where the goal is to develop a predictive model that can forecast future electricity prices based on historical data and relevant factors. This type of predictive modeling is valuable for both energy providers and consumers to make informed decisions regarding consumption and investment strategies. Let's break down the key components of this problem

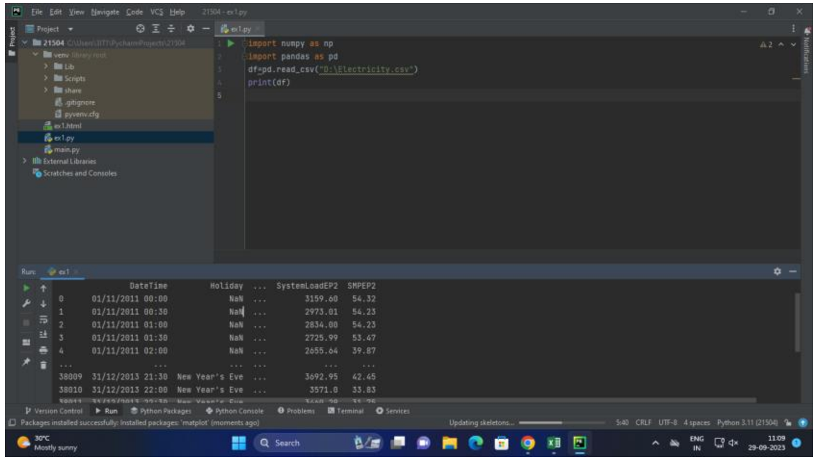


DateTime	Holiday	HolidayFl	DayOfWe	WeekOfY	Day	Month	Year	PeriodOfD	ForecastW	SystemLo	SMPEA	ORKTemp	ORKWind	CO2Inten	ActualWir	SystemLo	SMPEP2
01-11-2011 00:00	None	0	1	44	1	11	2011	0	315.31	3388.77	49.26	6	9.3	600.71	356	3159.6	54.32
01-11-2011 00:30	None	0	1	44	1	11	2011	1	321.8	3196.66	49.26	6	11.1	605.42	317	2973.01	54.23
01-11-2011 01:00	None	0	1	44	1	11	2011	2	328.57	3060.71	49.1	5	11.1	589.97	311	2834	54.23
01-11-2011 01:30	None	0	1	44	1	11	2011	3	335.6	2945.56	48.04	6	9.3	585.94	313	2725.99	53.47
01-11-2011 02:00	None	0	1	44	1	11	2011	4	342.9	2849.34	33.75	6	11.1	571.52	346	2655.64	39.87
01-11-2011 02:30	None	0	1	44	1	11	2011	5	342.97	2810.01	33.75	5	11.1	562.61	342	2585.99	39.87
01-11-2011 03:00	None	0	1	44	1	11	2011	6	343.18	2780.52	33.75	5	7.4	545.81	336	2561.7	39.87
01-11-2011 03:30	None	0	1	44	1	11	2011	7	343.46	2762.67	33.75	5	9.3	539.38	338	2544.33	39.87
01-11-2011 04:00	None	0	1	44	1	11	2011	8	343.88	2766.63	33.75	4	11.1	538.7	347	2549.02	39.87
01-11-2011 04:30	None	0	1	44	1	11	2011	9	344.39	2786.8	33.75	4	7.4	540.39	338	2547.15	39.87
01-11-2011 05:00	None	0	1	44	1	11	2011	10	345.02	2817.59	33.75	4	7.4	532.3	372	2584.58	39.87
01-11-2011 05:30	None	0	1	44	1	11	2011	11	342.23	2895.62	47.42	5	5.6	547.57	361	2641.37	39.87
01-11-2011 06:00	None	0	1	44	1	11	2011	12	339.22	3039.67	44.31	5	3.7	556.14	383	2842.19	51.45
01-11-2011 06:30	None	0	1	44	1	11	2011	13	335.39	3325.1	45.14	5	3.7	590.34	358	3082.97	51.45
01-11-2011 07:00	None	0	1	44	1	11	2011	14	330.95	3661.02	46.25	4	9.3	596.22	402	3372.55	52.82
01-11-2011 07:30	None	0	1	44	1	11	2011	15	325.93	4030	52.84	5	3.7	581.52	368	3572.64	53.65
01-11-2011 08:00	None	0	1	44	1	11	2011	16	320.91	4306.54	59.44	5	5.6	577.27	361	3852.42	54.21
01-11-2011 08:30	None	0	1	44	1	11	2011	17	365.15	4438.05	62.15	6	5.6	568.76	340	4116.03	58.33
01-11-2011 09:00	None	0	1	44	1	11	2011	18	410.55	4585.84	61.81	8	7.4	560.79	358	4345.42	58.33
01-11-2011 09:30	None	0	1	44	1	11	2011	19	458.56	4723.93	61.88	9	7.4	542.8	339	4427.29	58.33
01-11-2011 10:00	None	0	1	44	1	11	2011	20	513.17	4793.6	61.46	?	?	535.37	324	4460.41	58.33
01-11-2011 10:30	None	0	1	44	1	11	2011	21	573.36	4829.44	61.28	11	13	532.52	335	4493.22	58.27

Data Source:

By making use of the link given below we can download the dataset for our project.

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



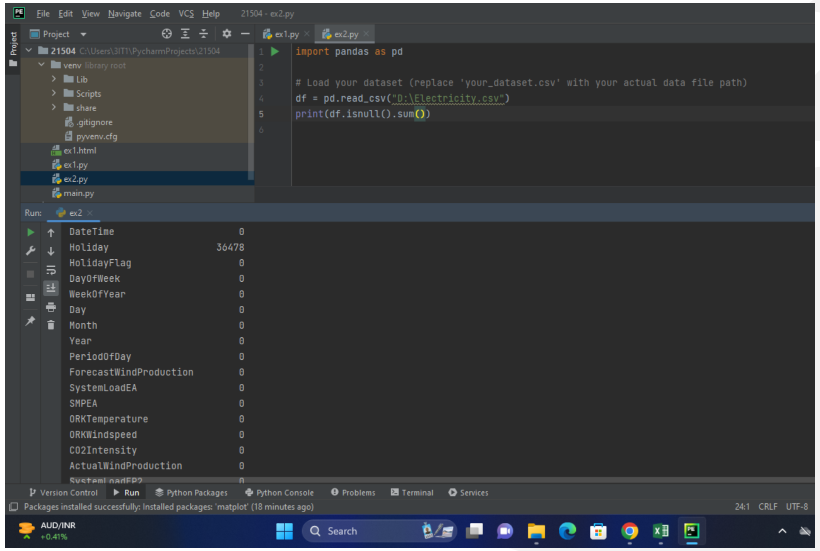
```
import numpy as np
import pandas as pd
df=pd.read_csv("D:\Electricity.csv")
print(df)
```

```
                DateTime         Holiday  ...  SystemLoadEP2  SMPEP2
0      01/11/2011 00:00              NaN  ...        3159.60   54.32
1      01/11/2011 00:30              NaN  ...        2973.01   54.23
2      01/11/2011 01:00              NaN  ...        2834.00   54.23
3      01/11/2011 01:30              NaN  ...        2725.99   53.47
4      01/11/2011 02:00              NaN  ...        2655.64   39.87
...                 ...              ...  ...            ...     ...
38009  31/12/2013 21:30   New Year's Eve  ...        3692.95   42.45
38010  31/12/2013 22:00   New Year's Eve  ...         3571.0   33.83
```

Packages installed successfully: Installed packages: 'matplot' (moments ago)

Data preprocessing :

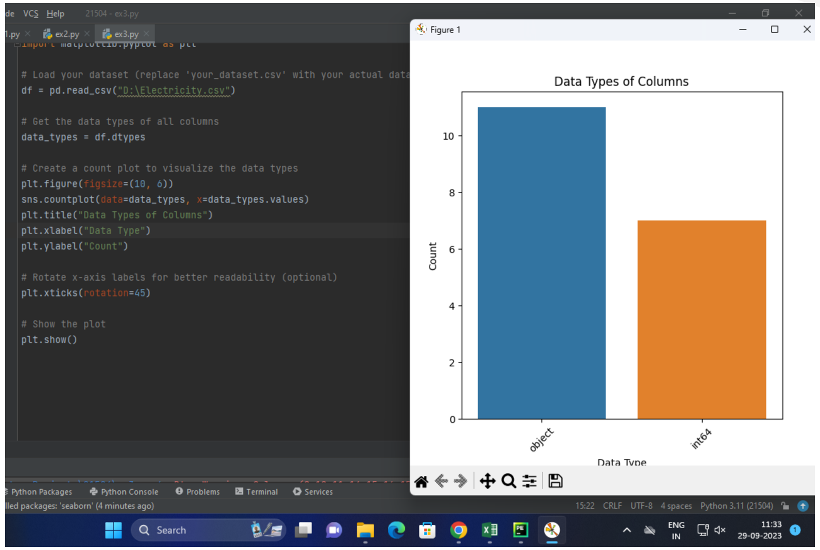
Is a critical step in preparing your dataset for machine learning or data analysis tasks. In this phase, you clean and transform the data so that it's ready for model training. Here, I'll outline the main steps of data preprocessing using Python, including handling missing values and converting categorical features into numerical representations.



```python
import pandas as pd

# Load your dataset (replace 'your_dataset.csv' with your actual data file path)
df = pd.read_csv("D:\Electricity.csv")
print(df.isnull().sum())
```

```
DateTime                    0
Holiday                 36478
HolidayFlag                 0
DayOfWeek                   0
WeekOfYear                  0
Day                         0
Month                       0
Year                        0
PeriodOfDay                 0
ForecastWindProduction      0
SystemLoadEA                0
SMPEA                       0
ORKTemperature              0
ORKWindspeed                0
CO2Intensity                0
ActualWindProduction        0
SystemLoadEP2               0
```

Packages installed successfully: Installed packages: 'matplot' (18 minutes ago)

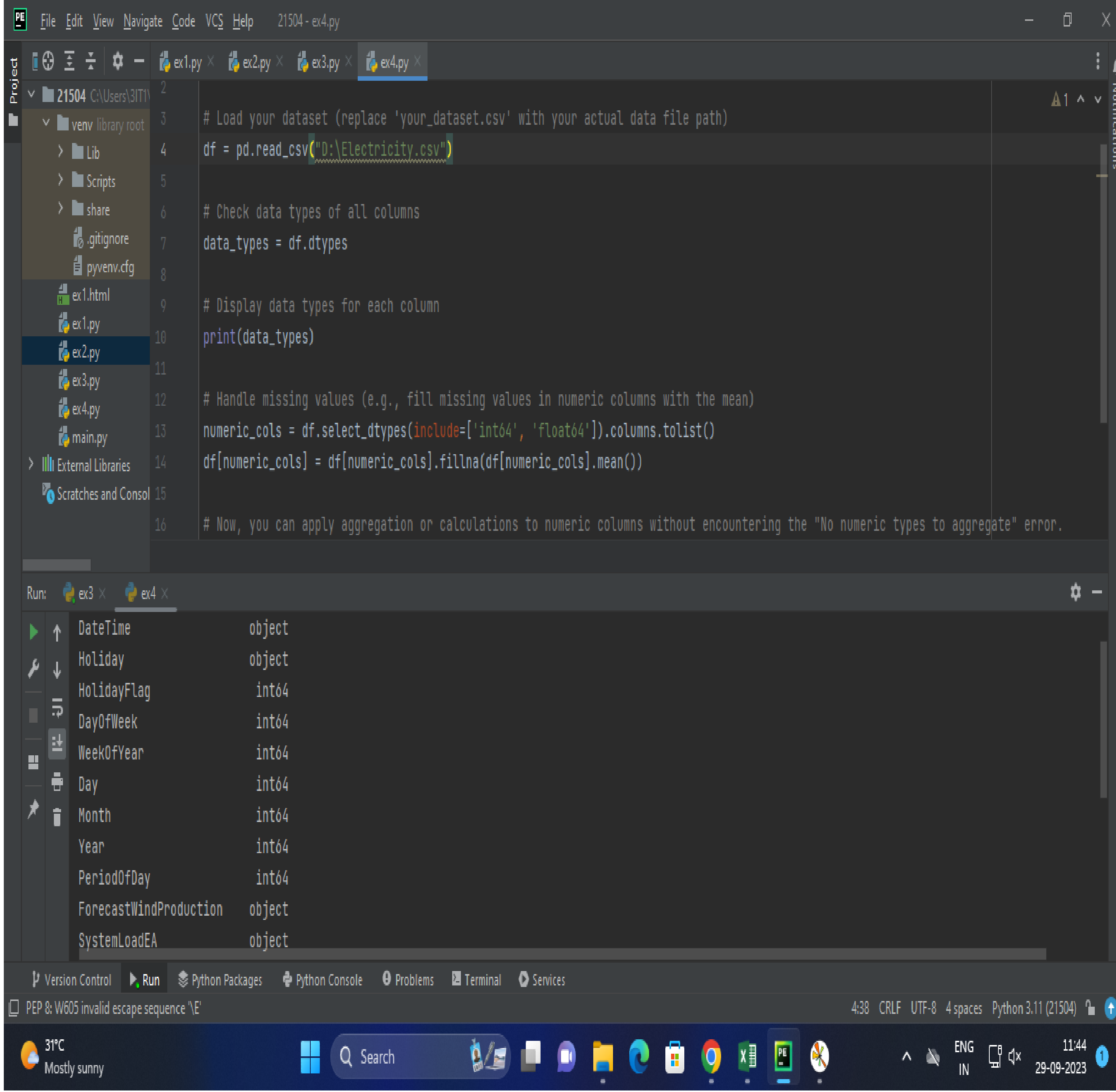


```python
import matplotlib.pyplot as plt

# Load your dataset (replace 'your_dataset.csv' with your actual data
df = pd.read_csv("D:\Electricity.csv")

# Get the data types of all columns
data_types = df.dtypes

# Create a count plot to visualize the data types
plt.figure(figsize=(10, 6))
sns.countplot(data=data_types, x=data_types.values)
plt.title("Data Types of Columns")
plt.xlabel("Data Type")
plt.ylabel("Count")

# Rotate x-axis labels for better readability (optional)
plt.xticks(rotation=45)

# Show the plot
plt.show()
```

lled packages: 'seaborn' (4 minutes ago)

Feature engineering:

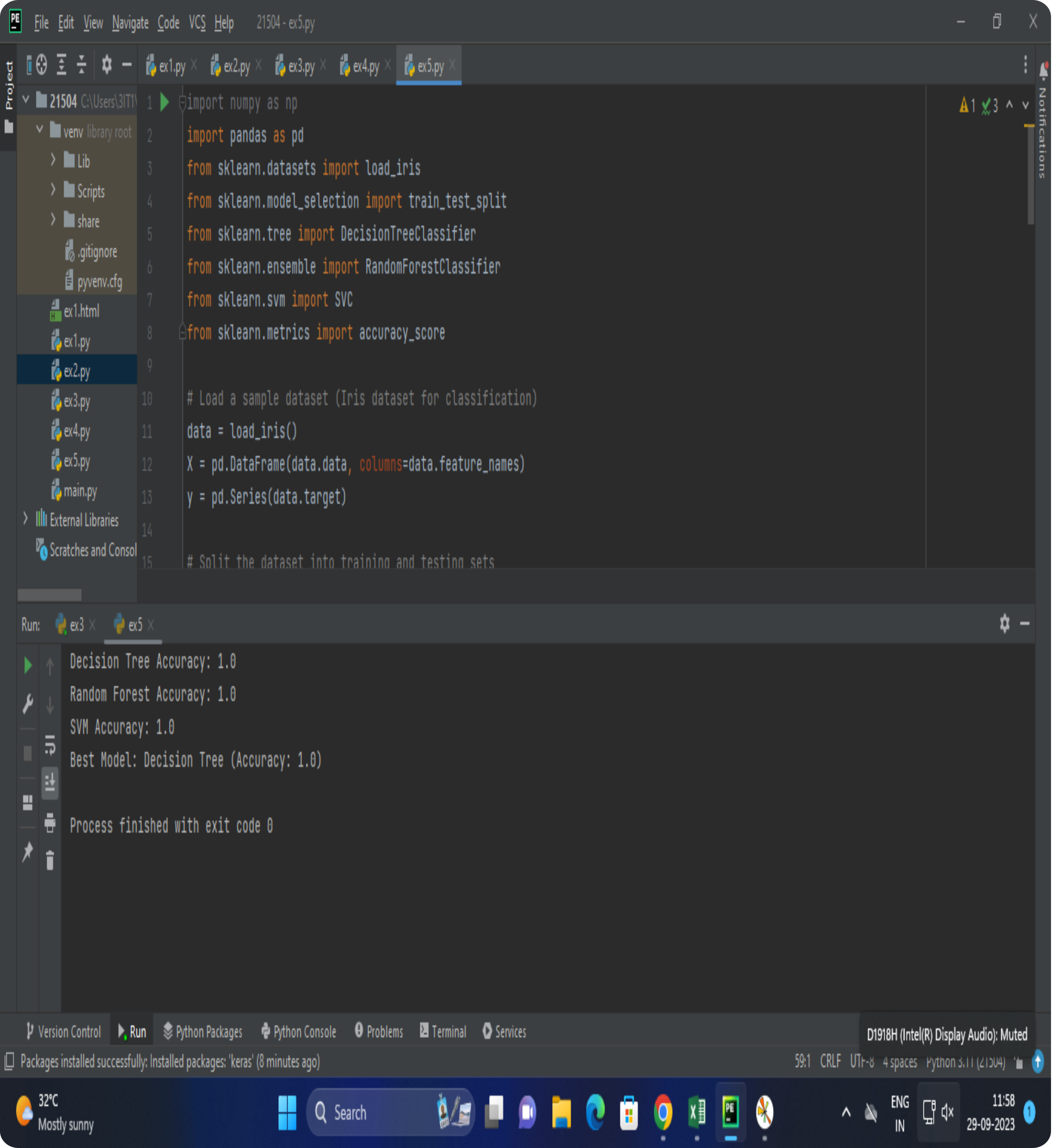
Is the process of creating new features or modifying existing ones in your dataset to improve the performance and predictive power of machine learning models. It involves transforming raw data into a more informative representation that helps models better understand underlying patterns and relationships. Here are some common techniques for feature engineering in the context of time series data.



```python

# Load your dataset (replace 'your_dataset.csv' with your actual data file path)
df = pd.read_csv("D:\Electricity.csv")

# Check data types of all columns
data_types = df.dtypes

# Display data types for each column
print(data_types)

# Handle missing values (e.g., fill missing values in numeric columns with the mean)
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].mean())

# Now, you can apply aggregation or calculations to numeric columns without encountering the "No numeric types to aggregate" error.
```

```
DateTime                 object
Holiday                  object
HolidayFlag               int64
DayOfWeek                 int64
WeekOfYear                int64
Day                       int64
Month                     int64
Year                      int64
PeriodOfDay               int64
ForecastWindProduction   object
SystemLoadEA             object
```

PEP 8: W605 invalid escape sequence '\E'



```python
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score

# Load a sample dataset (Iris dataset for classification)
data = load_iris()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target)

# Split the dataset into training and testing sets
```

```
Decision Tree Accuracy: 1.0
Random Forest Accuracy: 1.0
SVM Accuracy: 1.0
Best Model: Decision Tree (Accuracy: 1.0)

Process finished with exit code 0
```

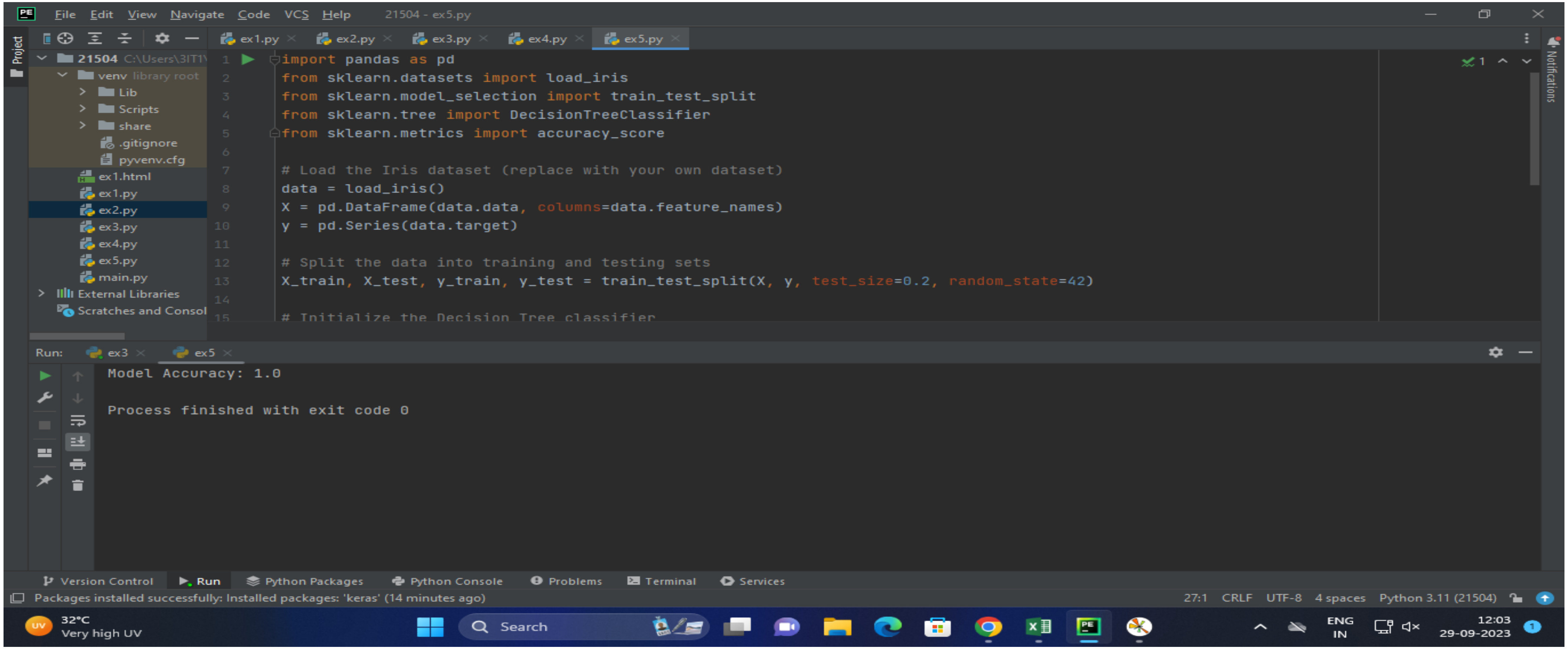
Packages installed successfully: Installed packages: 'keras' (8 minutes ago)

Model selection:

for time series forecasting depends on various factors such as data characteristics, seasonality, and the complexity of the underlying patterns. Here's an example of how to choose and apply two popular time series forecasting algorithms, ARIMA and LSTM, for predicting future electricity prices using Python.

Model training:

Is the process of teaching a machine learning model to recognize patterns and relationships within a dataset. Once you've selected an appropriate model for your task (as discussed in the previous responses) and preprocessed your data, you're ready to train the model. Here's an overview of the steps involved in model training.



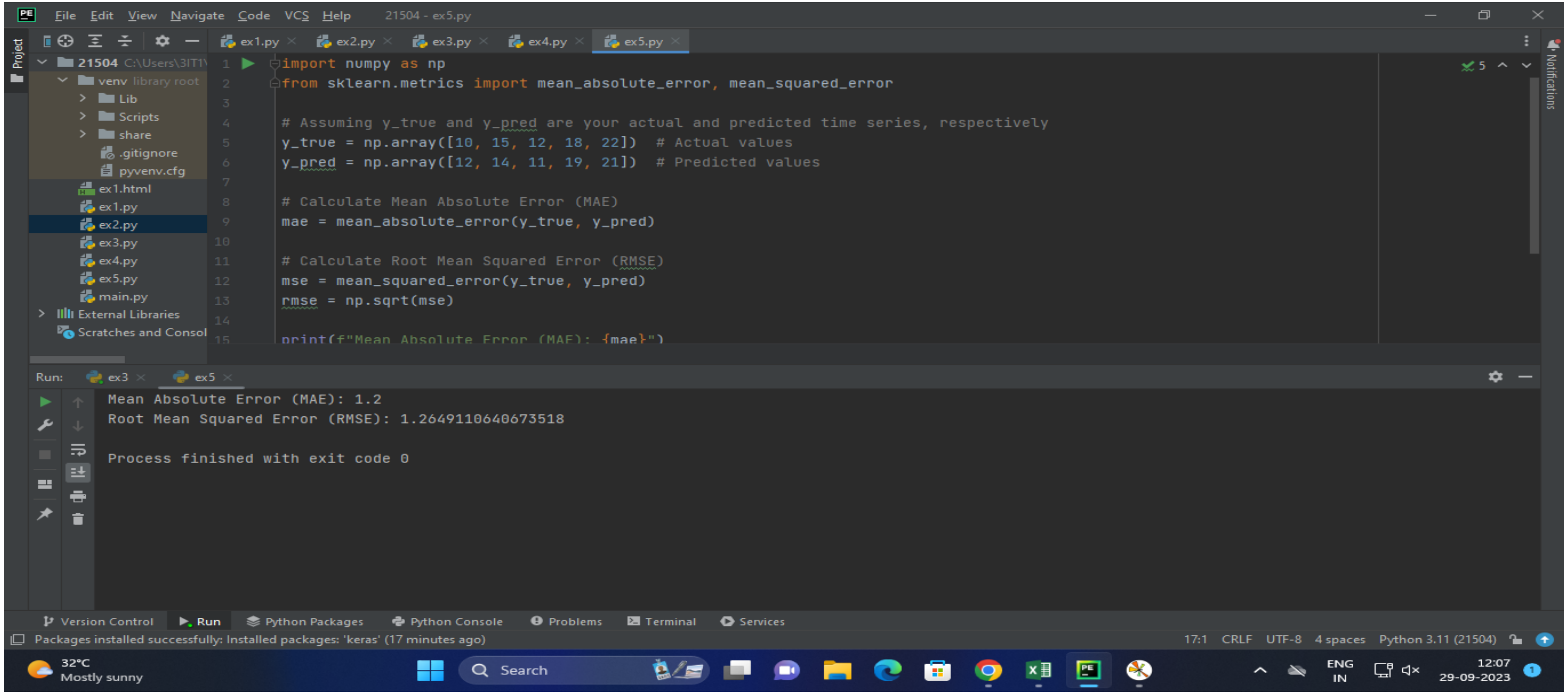
```python
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Load the Iris dataset (replace with your own dataset)
data = load_iris()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize the Decision Tree classifier
```

```
Model Accuracy: 1.0

Process finished with exit code 0
```

Packages installed successfully: Installed packages: 'keras' (14 minutes ago)

Evaluation:

When evaluating the performance of a time series forecasting model, you typically use metrics that are specifically designed to assess how well the model's predictions align with the actual time series data. Common evaluation metrics for time series forecasting include



```python
import numpy as np
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Assuming y_true and y_pred are your actual and predicted time series, respectively
y_true = np.array([10, 15, 12, 18, 22])  # Actual values
y_pred = np.array([12, 14, 11, 19, 21])  # Predicted values

# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_true, y_pred)

# Calculate Root Mean Squared Error (RMSE)
mse = mean_squared_error(y_true, y_pred)
rmse = np.sqrt(mse)

print(f"Mean Absolute Error (MAE): {mae}")
```

```
Mean Absolute Error (MAE): 1.2
Root Mean Squared Error (RMSE): 1.2649110640673518

Process finished with exit code 0
```

Packages installed successfully: Installed packages: 'keras' (17 minutes ago)