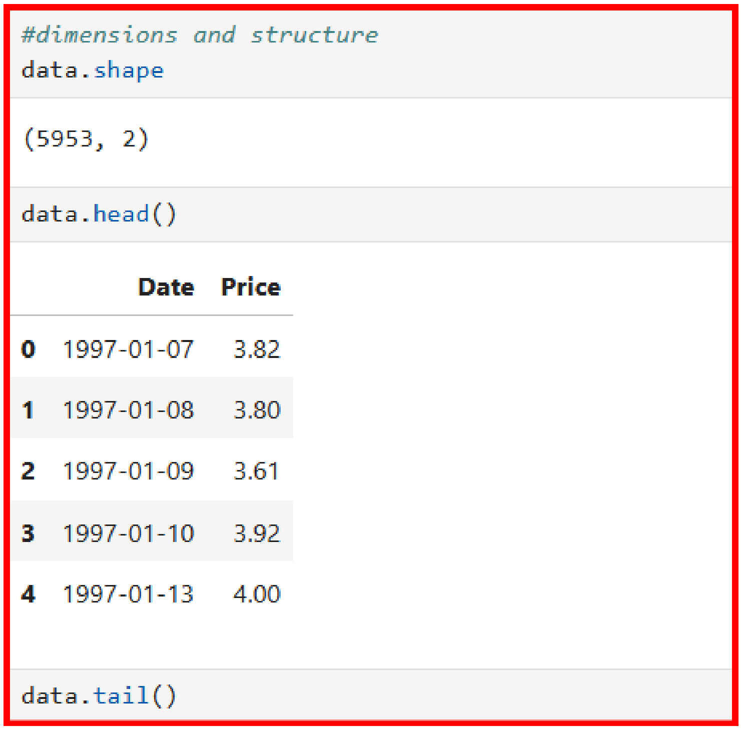
Data Collection and Preprocessing Phase

Date	8 July 2024
Team ID	SWTID1720096271
Project Title	Machine learning approach for Predicting the price of natural gas
Maximum Marks	6 Marks

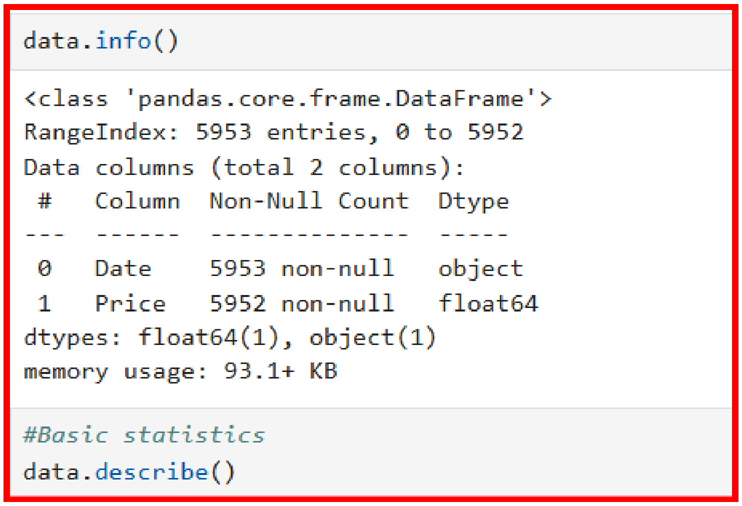
Data Exploration and Preprocessing Template

Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

Section	**Description**

Data Overview	
	`#dimensions and structure`
	`data.shape`
	`(5953, 2)`
	`data.head()`
	`   Date        Price`
	`0  1997-01-07  3.82`
	`1  1997-01-08  3.80`
	`2  1997-01-09  3.61`
	`3  1997-01-10  3.92`
	`4  1997-01-13  4.00`
	`data.tail()`

<table>
<tr>
<td></td>
<td>

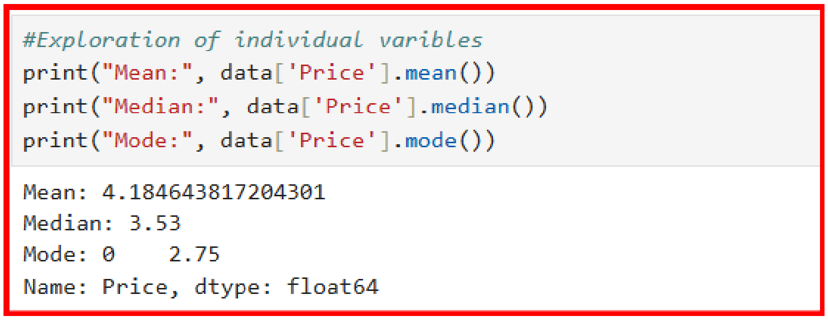


```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5953 entries, 0 to 5952
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Date    5953 non-null   object 
 1   Price   5952 non-null   float64
dtypes: float64(1), object(1)
memory usage: 93.1+ KB
```

```
#Basic statistics
data.describe()
```

</td>
</tr>
<tr>
<td>Univariate Analysis</td>
<td>

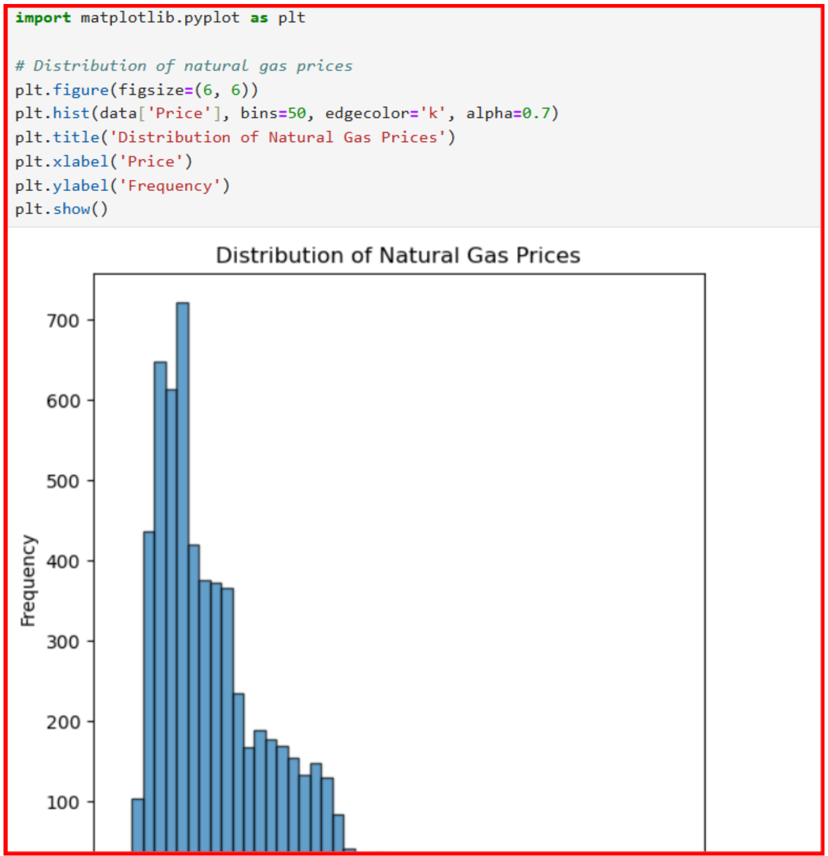


```
#Exploration of individual varibles
print("Mean:", data['Price'].mean())
print("Median:", data['Price'].median())
print("Mode:", data['Price'].mode())
```

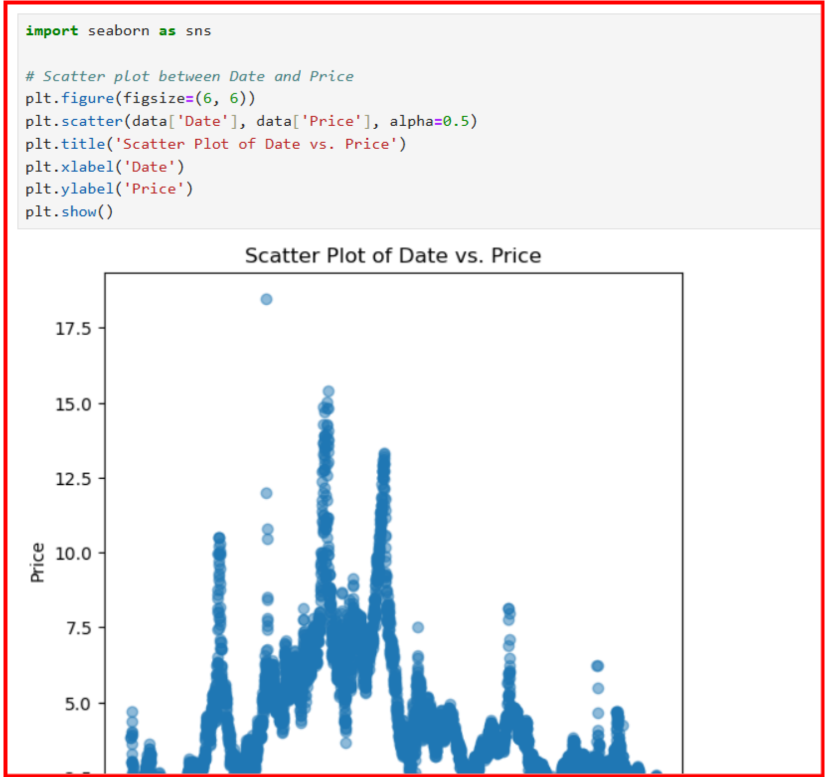
```
Mean: 4.184643817204301
Median: 3.53
Mode: 0    2.75
Name: Price, dtype: float64
```

</td>
</tr>
</table>

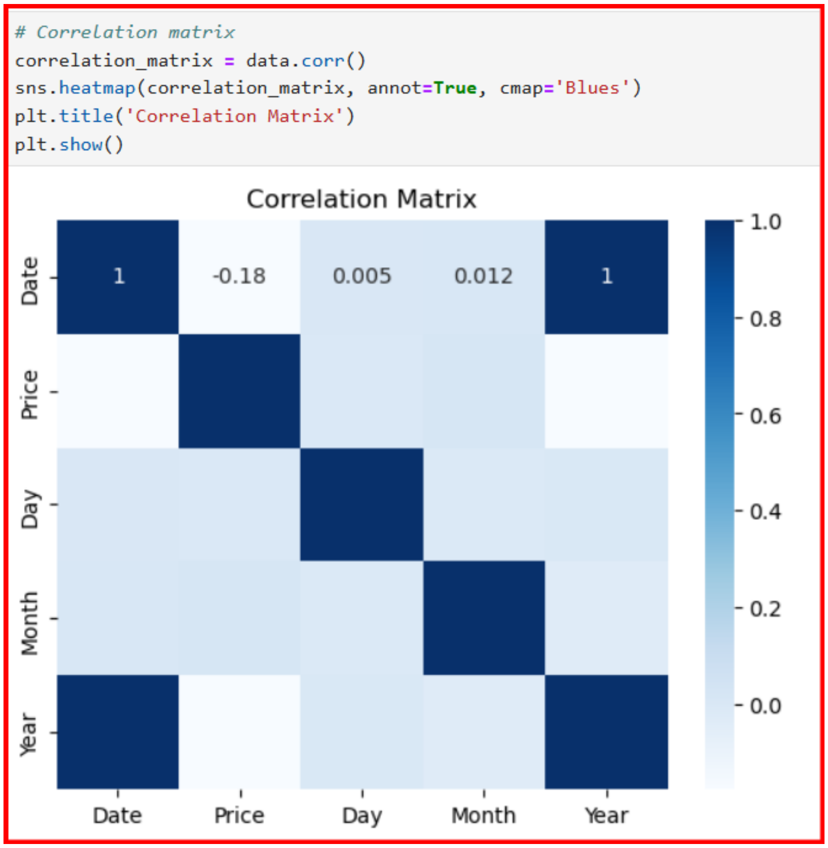
Bivariate Analysis

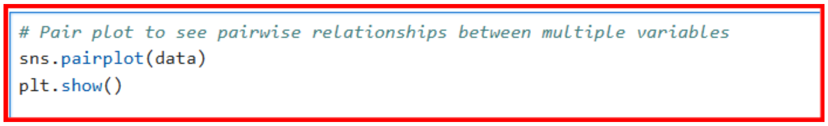


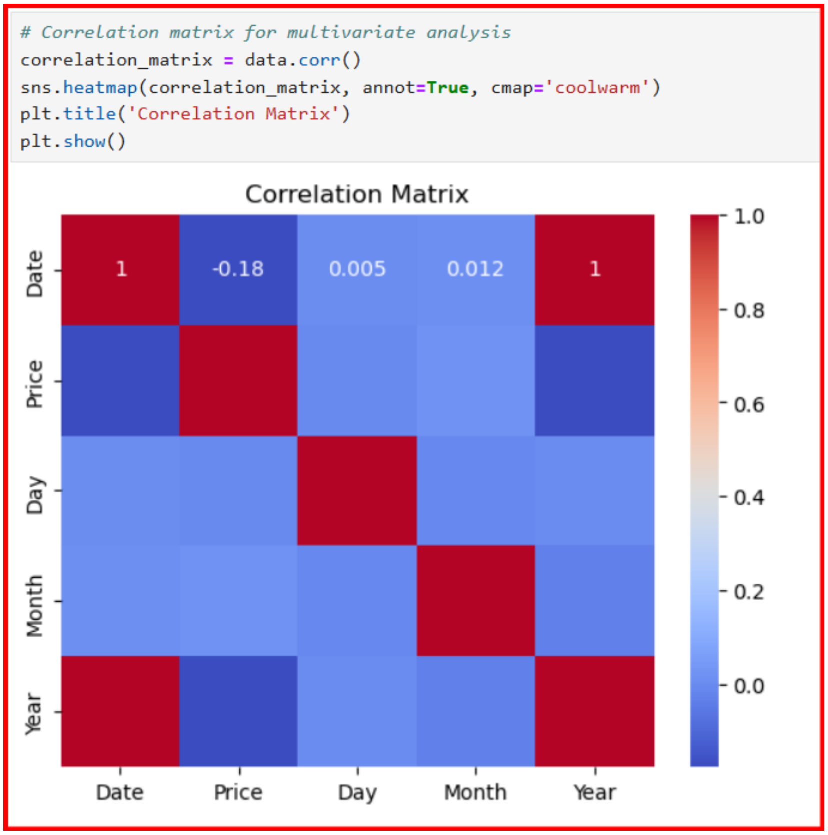
```
import matplotlib.pyplot as plt

# Distribution of natural gas prices
plt.figure(figsize=(6, 6))
plt.hist(data['Price'], bins=50, edgecolor='k', alpha=0.7)
plt.title('Distribution of Natural Gas Prices')
plt.xlabel('Price')
plt.ylabel('Frequency')
plt.show()
```



```
import seaborn as sns

# Scatter plot between Date and Price
plt.figure(figsize=(6, 6))
plt.scatter(data['Date'], data['Price'], alpha=0.5)
plt.title('Scatter Plot of Date vs. Price')
plt.xlabel('Date')
plt.ylabel('Price')
plt.show()
```

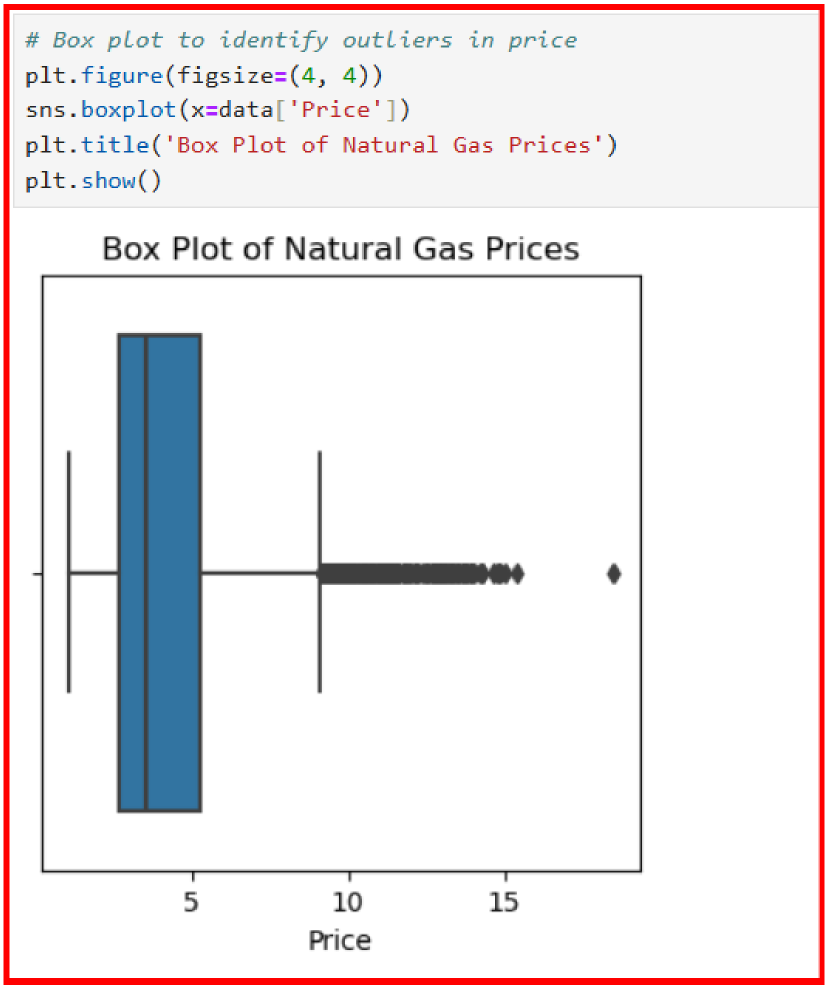
	
	`# Correlation matrix`
	`correlation_matrix = data.corr()`
	`sns.heatmap(correlation_matrix, annot=True, cmap='Blues')`
	`plt.title('Correlation Matrix')`
	`plt.show()`
	 Correlation Matrix
	1
	-0.18
	0.005
	0.012
	1
	Date
	Price
	Day
	Month
	Year
	Date
	Price
	Day
	Month
	Year
	1.0
	0.8
	0.6
	0.4
	0.2
	0.0

Multivariate Analysis	
	`# Pair plot to see pairwise relationships between multiple variables`
	`sns.pairplot(data)`
	`plt.show()`

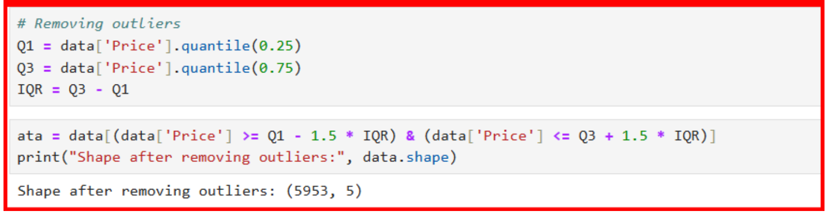


```
# Correlation matrix for multivariate analysis
correlation_matrix = data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

Outliers and Anomalies



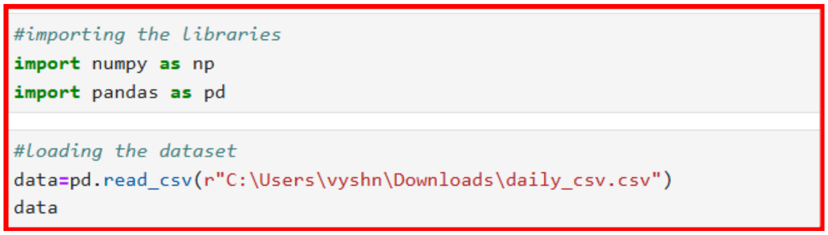
```
# Box plot to identify outliers in price
plt.figure(figsize=(4, 4))
sns.boxplot(x=data['Price'])
plt.title('Box Plot of Natural Gas Prices')
plt.show()
```



```
# Removing outliers
Q1 = data['Price'].quantile(0.25)
Q3 = data['Price'].quantile(0.75)
IQR = Q3 - Q1

ata = data[(data['Price'] >= Q1 - 1.5 * IQR) & (data['Price'] <= Q3 + 1.5 * IQR)]
print("Shape after removing outliers:", data.shape)

Shape after removing outliers: (5953, 5)
```

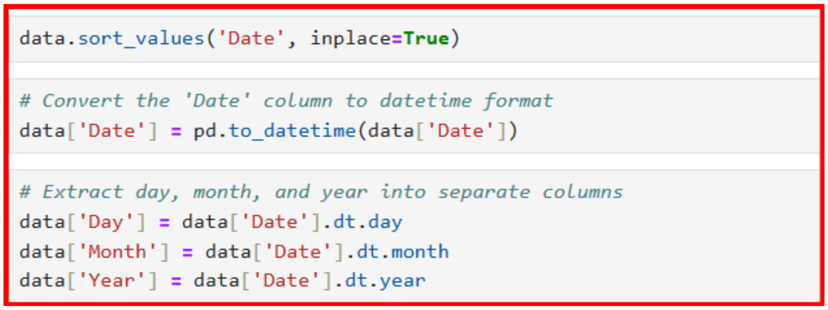
Data Preprocessing Code Screenshots

Loading Data



```
#importing the libraries
import numpy as np
import pandas as pd

#loading the dataset
data=pd.read_csv(r"C:\Users\vyshn\Downloads\daily_csv.csv")
data
```

<table>
<tr>
<td></td>
<td>

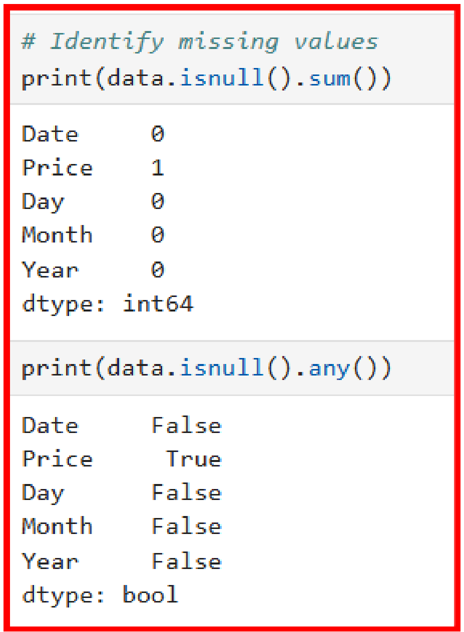


```
data.sort_values('Date', inplace=True)
```

```
# Convert the 'Date' column to datetime format
data['Date'] = pd.to_datetime(data['Date'])
```

```
# Extract day, month, and year into separate columns
data['Day'] = data['Date'].dt.day
data['Month'] = data['Date'].dt.month
data['Year'] = data['Date'].dt.year
```

</td>
</tr>
<tr>
<td>Handling Missing Data</td>
<td>

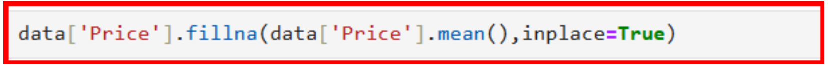


```
# Identify missing values
print(data.isnull().sum())
```

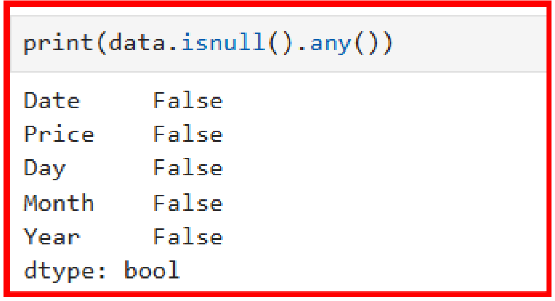
```
Date      0
Price     1
Day       0
Month     0
Year      0
dtype: int64
```

```
print(data.isnull().any())
```

```
Date      False
Price      True
Day       False
Month     False
Year      False
dtype: bool
```



```
data['Price'].fillna(data['Price'].mean(),inplace=True)
```

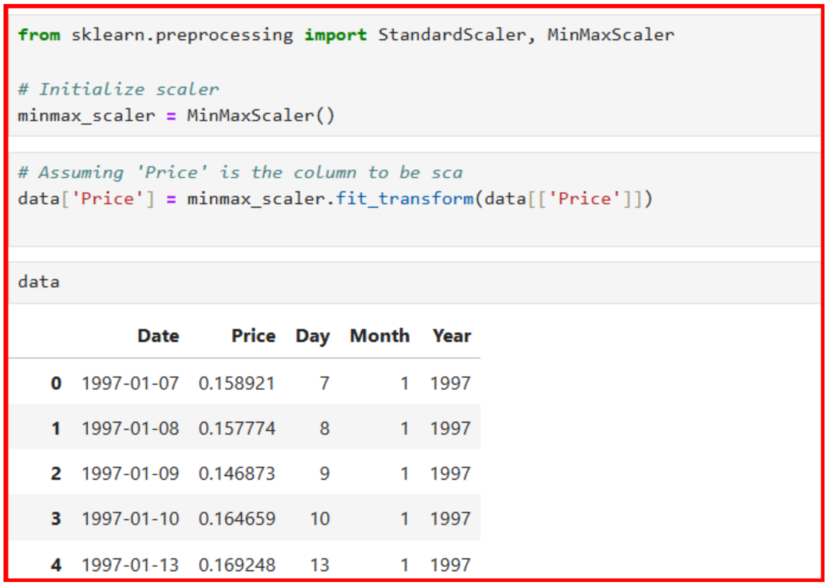


```
print(data.isnull().any())
```

```
Date      False
Price     False
Day       False
Month     False
Year      False
dtype: bool
```

</td>
</tr>
</table>

Data Transformation



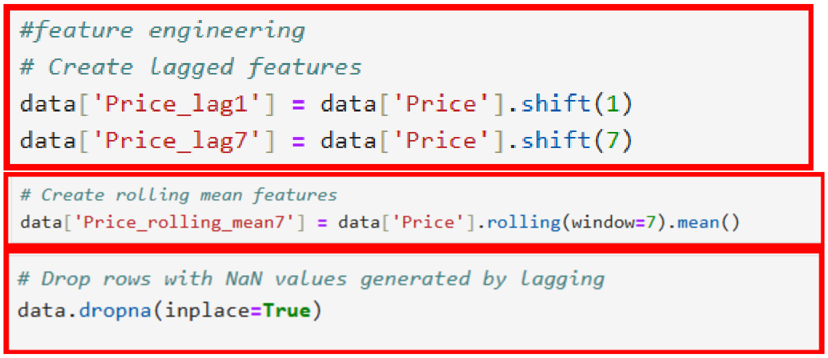
```
from sklearn.preprocessing import StandardScaler, MinMaxScaler

# Initialize scaler
minmax_scaler = MinMaxScaler()
```

```
# Assuming 'Price' is the column to be sca
data['Price'] = minmax_scaler.fit_transform(data[['Price']])
```

```
data
```

```
         Date     Price  Day  Month  Year
0  1997-01-07  0.158921    7      1  1997
1  1997-01-08  0.157774    8      1  1997
2  1997-01-09  0.146873    9      1  1997
3  1997-01-10  0.164659   10      1  1997
4  1997-01-13  0.169248   13      1  1997
```

Feature Engineering

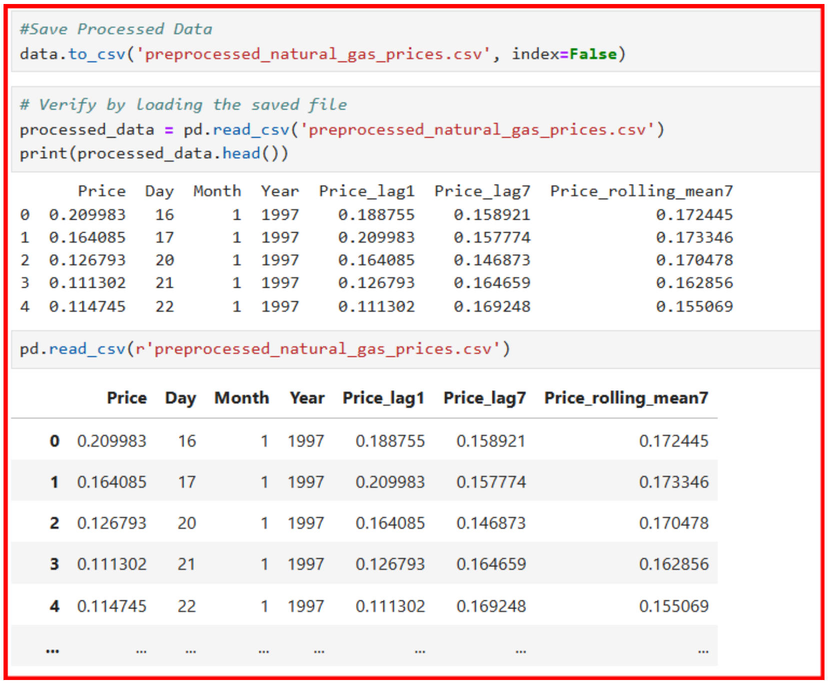


```
#feature engineering
# Create lagged features
data['Price_lag1'] = data['Price'].shift(1)
data['Price_lag7'] = data['Price'].shift(7)
```

```
# Create rolling mean features
data['Price_rolling_mean7'] = data['Price'].rolling(window=7).mean()
```

```
# Drop rows with NaN values generated by lagging
data.dropna(inplace=True)
```

Save Processed Data



```
#Save Processed Data
data.to_csv('preprocessed_natural_gas_prices.csv', index=False)
```

```
# Verify by loading the saved file
processed_data = pd.read_csv('preprocessed_natural_gas_prices.csv')
print(processed_data.head())
```

```
      Price  Day  Month  Year  Price_lag1  Price_lag7  Price_rolling_mean7
0  0.209983   16      1  1997    0.188755    0.158921             0.172445
1  0.164085   17      1  1997    0.209983    0.157774             0.173346
2  0.126793   20      1  1997    0.164085    0.146873             0.170478
3  0.111302   21      1  1997    0.126793    0.164659             0.162856
4  0.114745   22      1  1997    0.111302    0.169248             0.155069
```

```
pd.read_csv(r'preprocessed_natural_gas_prices.csv')
```

```
       Price  Day  Month  Year  Price_lag1  Price_lag7  Price_rolling_mean7
0   0.209983   16      1  1997    0.188755    0.158921             0.172445
1   0.164085   17      1  1997    0.209983    0.157774             0.173346
2   0.126793   20      1  1997    0.164085    0.146873             0.170478
3   0.111302   21      1  1997    0.126793    0.164659             0.162856
4   0.114745   22      1  1997    0.111302    0.169248             0.155069
...      ...  ...    ...   ...         ...         ...                  ...
```