Hypothesis Testing

In [2]:
```
import pandas as pd
import numpy as np
from scipy.stats import f_oneway

# Load the datasets
pjme_data = pd.read_csv('PJME_hourly.csv')
pjmw_data = pd.read_csv('PJMW_hourly.csv')

# Convert 'Datetime' columns to datetime type
pjme_data['Datetime'] = pd.to_datetime(pjme_data['Datetime'])
pjmw_data['Datetime'] = pd.to_datetime(pjmw_data['Datetime'])

# Merge the datasets on 'Datetime'
merged_data = pd.merge(pjme_data, pjmw_data, on='Datetime', how='inner')

# Function to categorize months into seasons
def get_season(month):
    if month in [12, 1, 2]:
        return 'Winter'
    elif month in [3, 4, 5]:
        return 'Spring'
    elif month in [6, 7, 8]:
        return 'Summer'
    elif month in [9, 10, 11]:
        return 'Autumn'

# Add 'Season' column based on 'Datetime'
merged_data['Season'] = merged_data['Datetime'].dt.month.apply(get_season)

# Group data by season and calculate mean for PJME and PJMW
seasonal_data = merged_data.groupby('Season').agg({'PJME_MW': 'mean', 'PJMW_MW': 'mean'}

# Display the seasonal data
print(seasonal_data)

print("\n")

# Perform ANOVA test for PJME and PJMW
anova_pjme = f_oneway(
    merged_data[merged_data['Season'] == 'Winter']['PJME_MW'],
    merged_data[merged_data['Season'] == 'Spring']['PJME_MW'],
    merged_data[merged_data['Season'] == 'Summer']['PJME_MW'],
    merged_data[merged_data['Season'] == 'Autumn']['PJME_MW']
)

anova_pjmw = f_oneway(
    merged_data[merged_data['Season'] == 'Winter']['PJMW_MW'],
    merged_data[merged_data['Season'] == 'Spring']['PJMW_MW'],
    merged_data[merged_data['Season'] == 'Summer']['PJMW_MW'],
    merged_data[merged_data['Season'] == 'Autumn']['PJMW_MW']
)

# Print ANOVA results
print('ANOVA result for PJME:', anova_pjme)
print('ANOVA result for PJMW:', anova_pjmw)
```

```
             PJME_MW      PJMW_MW
Season
Autumn  29625.682721  5199.901929
Spring  29040.273400  5224.394790
Summer  36112.459515  5734.206129
Winter  33618.397057  6268.813851
```

```
ANOVA result for PJME: F_onewayResult(statistic=12249.68011422396, pvalue=0.0)
ANOVA result for PJMW: F_onewayResult(statistic=11655.180526622571, pvalue=0.0)
```

Correlation Analysis

In [3]:

```python
import pandas as pd
from scipy.stats import pearsonr

# Load the datasets
pjme_data = pd.read_csv('PJME_hourly.csv')
pjmw_data = pd.read_csv('PJMW_hourly.csv')

# Convert 'Datetime' columns to datetime type
pjme_data['Datetime'] = pd.to_datetime(pjme_data['Datetime'])
pjmw_data['Datetime'] = pd.to_datetime(pjmw_data['Datetime'])

# Merge the datasets on 'Datetime'
merged_data = pd.merge(pjme_data, pjmw_data, on='Datetime', how='inner')

# Perform Pearson correlation test
correlation_coefficient, p_value = pearsonr(merged_data['PJME_MW'], merged_data['PJMW_MW

# Print the correlation coefficient and p-value
print("Correlation Coefficient:", correlation_coefficient)
print("P-value:", p_value)
```

```
Correlation Coefficient: 0.8757346767499891
P-value: 0.0
```

Anova

In [5]:

```python
import pandas as pd
from scipy.stats import f_oneway

# Load the datasets
pjme_data = pd.read_csv('PJME_hourly.csv')
pjmw_data = pd.read_csv('PJMW_hourly.csv')

# Ensure the 'PJME_MW' and 'PJMW_MW' columns are correctly named and used here
# Perform ANOVA to compare the average hourly electricity consumption in PJME and PJMW
anova_result = f_oneway(pjme_data['PJME_MW'], pjmw_data['PJMW_MW'])

# Print the ANOVA results: F-statistic and p-value
print("ANOVA F-statistic:", anova_result.statistic)
print("ANOVA p-value:", anova_result.pvalue)

# Based on the p-value, conclude if there is a significant difference or not
if anova_result.pvalue < 0.05:
    print("Reject the null hypothesis: There is a significant difference between the ave
else:
    print("Fail to reject the null hypothesis: There is no significant difference betwee
```

```
ANOVA F-statistic: 2349712.4439348853
ANOVA p-value: 0.0
Reject the null hypothesis: There is a significant difference between the average hourly
electricity consumption in PJME and PJMW.
```

Linear regression

In [6]:

```python
import pandas as pd
import statsmodels.api as sm

# Load the datasets
```

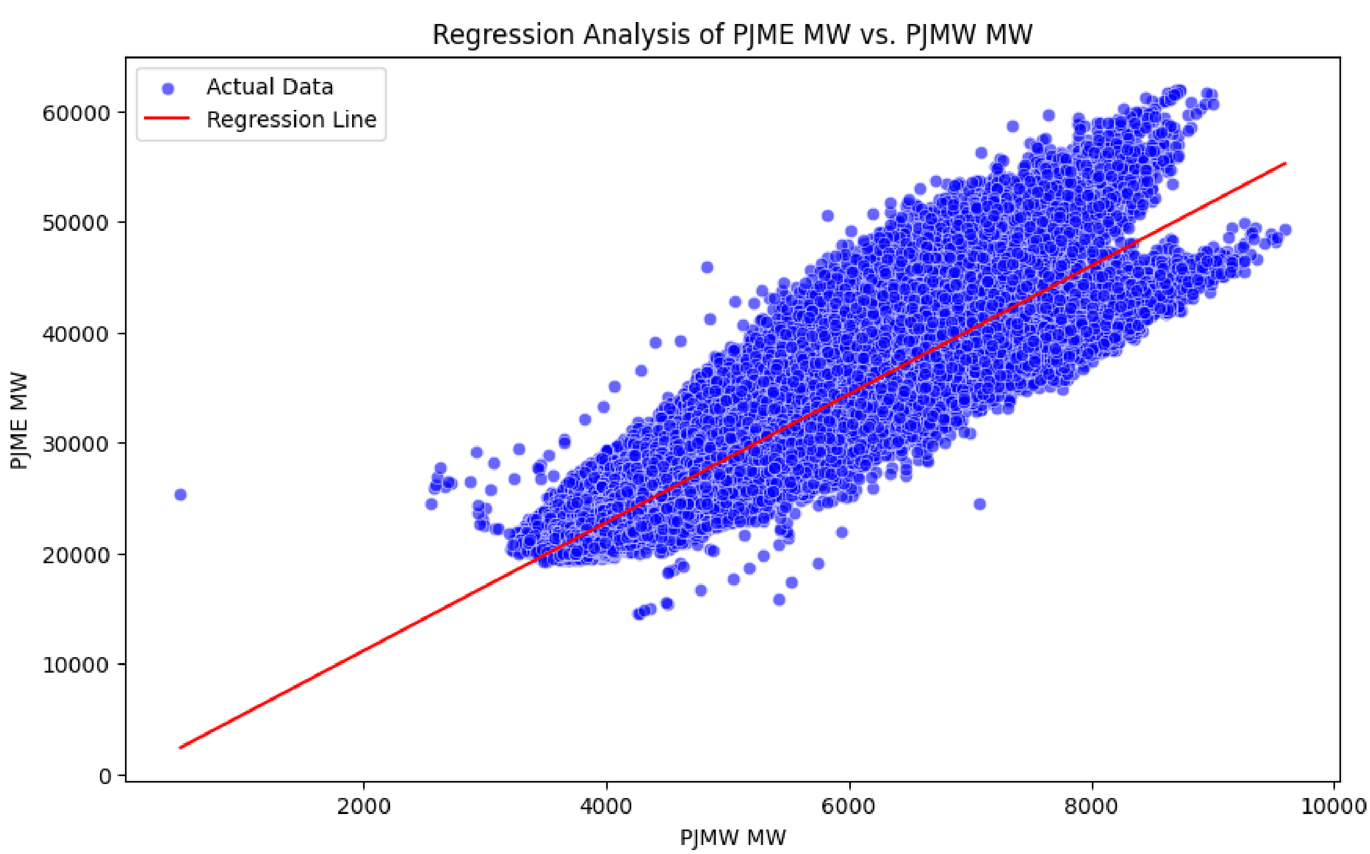
```python
pjme_data = pd.read_csv('PJME_hourly.csv')
pjmw_data = pd.read_csv('PJMW_hourly.csv')

# It's assumed both datasets are aligned by the same datetime, hence merging is required
# Make sure both datasets have the 'Datetime' column for a proper merge
pjme_data['Datetime'] = pd.to_datetime(pjme_data['Datetime'])
pjmw_data['Datetime'] = pd.to_datetime(pjmw_data['Datetime'])

# Merge datasets on 'Datetime'
data_merged = pd.merge(pjme_data, pjmw_data, on='Datetime')

# Check the merged data
print(data_merged.head())

# Set up the dependent variable (y) and independent variable (x)
# Assuming 'PJME_MW' is the dependent variable and 'PJMW_MW' the independent
X = data_merged['PJMW_MW']  # Independent variable
y = data_merged['PJME_MW']  # Dependent variable

# Add a constant to the model (the intercept)
X = sm.add_constant(X)

# Create a model and fit it
model = sm.OLS(y, X).fit()

# Print out the statistics
print(model.summary())
```

```
             Datetime  PJME_MW  PJMW_MW
0 2002-12-31 01:00:00  26498.0   5077.0
1 2002-12-31 02:00:00  25147.0   4939.0
2 2002-12-31 03:00:00  24574.0   4885.0
3 2002-12-31 04:00:00  24393.0   4857.0
4 2002-12-31 05:00:00  24860.0   4930.0
                            OLS Regression Results
==============================================================================
Dep. Variable:                PJME_MW   R-squared:                       0.767
Model:                            OLS   Adj. R-squared:                  0.767
Method:                 Least Squares   F-statistic:                 4.712e+05
Date:                Thu, 02 May 2024   Prob (F-statistic):               0.00
Time:                        02:48:32   Log-Likelihood:            -1.3560e+06
No. Observations:              143214   AIC:                         2.712e+06
Df Residuals:                  143212   BIC:                         2.712e+06
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -399.9171     48.079     -8.318      0.000    -494.150    -305.684
PJMW_MW        5.8030      0.008    686.438      0.000       5.786       5.820
==============================================================================
Omnibus:                    12343.955   Durbin-Watson:                   0.033
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16419.129
Skew:                           0.739   Prob(JB):                         0.00
Kurtosis:                       3.754   Cond. No.                     3.30e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
ed.
[2] The condition number is large, 3.3e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [7]:

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
import statsmodels.api as sm

# Load the datasets
pjme_data = pd.read_csv('PJME_hourly.csv')
pjmw_data = pd.read_csv('PJMW_hourly.csv')

# Convert 'Datetime' columns to datetime type for proper alignment
pjme_data['Datetime'] = pd.to_datetime(pjme_data['Datetime'])
pjmw_data['Datetime'] = pd.to_datetime(pjmw_data['Datetime'])

# Merge the datasets on 'Datetime'
data_merged = pd.merge(pjme_data, pjmw_data, on='Datetime')

# Set up the independent variable (X) and dependent variable (y)
X = data_merged['PJMW_MW']  # Independent variable
y = data_merged['PJME_MW']  # Dependent variable

# Add a constant to the model (the intercept)
X = sm.add_constant(X)

# Create a model and fit it
model = sm.OLS(y, X).fit()

# Predictions for plotting
data_merged['predicted'] = model.predict(X)

# Plotting
plt.figure(figsize=(10, 6))
sns.scatterplot(x='PJMW_MW', y='PJME_MW', data=data_merged, color='blue', alpha=0.6, lab
sns.lineplot(x='PJMW_MW', y='predicted', data=data_merged, color='red', label='Regressio
plt.title('Regression Analysis of PJME MW vs. PJMW MW')
plt.xlabel('PJMW MW')
plt.ylabel('PJME MW')
plt.legend()
plt.show()

# Print model summary
print(model.summary())
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                PJME_MW   R-squared:                       0.767
Model:                            OLS   Adj. R-squared:                  0.767
Method:                 Least Squares   F-statistic:                 4.712e+05
Date:                Thu, 02 May 2024   Prob (F-statistic):               0.00
Time:                        02:53:08   Log-Likelihood:            -1.3560e+06
No. Observations:              143214   AIC:                         2.712e+06
Df Residuals:                  143212   BIC:                         2.712e+06
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -399.9171     48.079     -8.318      0.000    -494.150    -305.684
PJMW_MW        5.8030      0.008    686.438      0.000       5.786       5.820
==============================================================================
Omnibus:                    12343.955   Durbin-Watson:                   0.033
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            16419.129
Skew:                           0.739   Prob(JB):                         0.00
Kurtosis:                       3.754   Cond. No.                     3.30e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
ed.
[2] The condition number is large, 3.3e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [ ]: