```python
import pandas as pd

# Load the Excel file into a pandas DataFrame
df = pd.read_excel('/content/ENB2012_data[1].xlsx')

# Display the first 5 rows of the DataFrame
print('First 5 rows of the DataFrame:')
print(df.head())

print('\nDataFrame Info (column names and data types):')
# Print the column names and their respective data types
df.info()
```

```
First 5 rows of the DataFrame:
     X1     X2     X3      X4   X5  X6   X7  X8     Y1     Y2
0  0.98  514.5  294.0  110.25  7.0   2  0.0   0  15.55  21.33
1  0.98  514.5  294.0  110.25  7.0   3  0.0   0  15.55  21.33
2  0.98  514.5  294.0  110.25  7.0   4  0.0   0  15.55  21.33
3  0.98  514.5  294.0  110.25  7.0   5  0.0   0  15.55  21.33
4  0.90  563.5  318.5  122.50  7.0   2  0.0   0  20.84  28.28

DataFrame Info (column names and data types):
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 10 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   X1      768 non-null    float64
 1   X2      768 non-null    float64
 2   X3      768 non-null    float64
 3   X4      768 non-null    float64
 4   X5      768 non-null    float64
 5   X6      768 non-null    int64
 6   X7      768 non-null    float64
 7   X8      768 non-null    int64
 8   Y1      768 non-null    float64
 9   Y2      768 non-null    float64
dtypes: float64(8), int64(2)
memory usage: 60.1 KB
```

```python
df = df.rename(columns={'Y1': 'Heating Load', 'Y2': 'Cooling Load'})

print('DataFrame after renaming Y1 and Y2 columns:')
print(df.head())
```

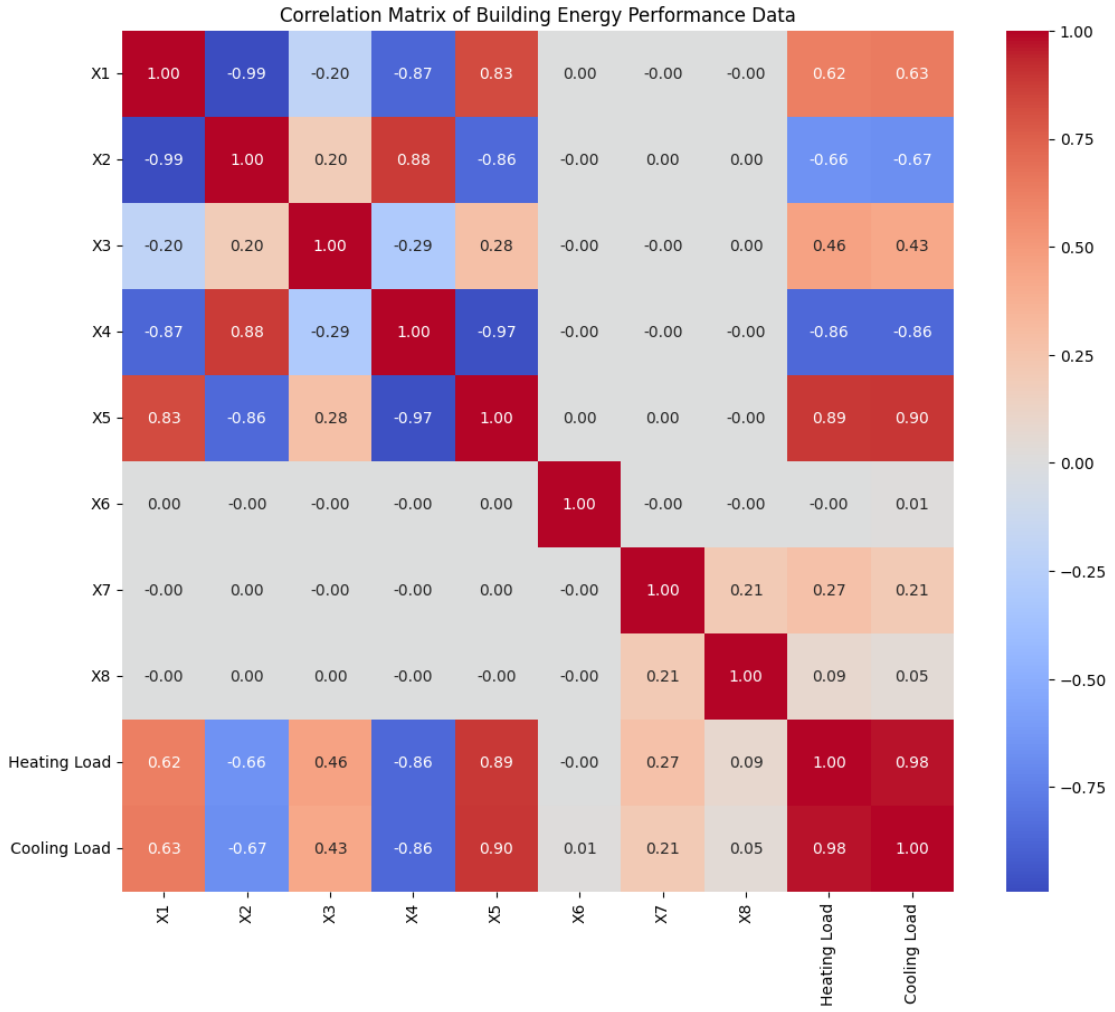
```
DataFrame after renaming Y1 and Y2 columns:
     X1     X2     X3      X4   X5  X6   X7  X8  Heating Load  Cooling Load
0  0.98  514.5  294.0  110.25  7.0   2  0.0   0         15.55         21.33
1  0.98  514.5  294.0  110.25  7.0   3  0.0   0         15.55         21.33
2  0.98  514.5  294.0  110.25  7.0   4  0.0   0         15.55         21.33
```

```
3  0.98  514.5  294.0  110.25  7.0   5  0.0   0         15.55         21.33
4  0.90  563.5  318.5  122.50  7.0   2  0.0   0         20.84         28.28
```

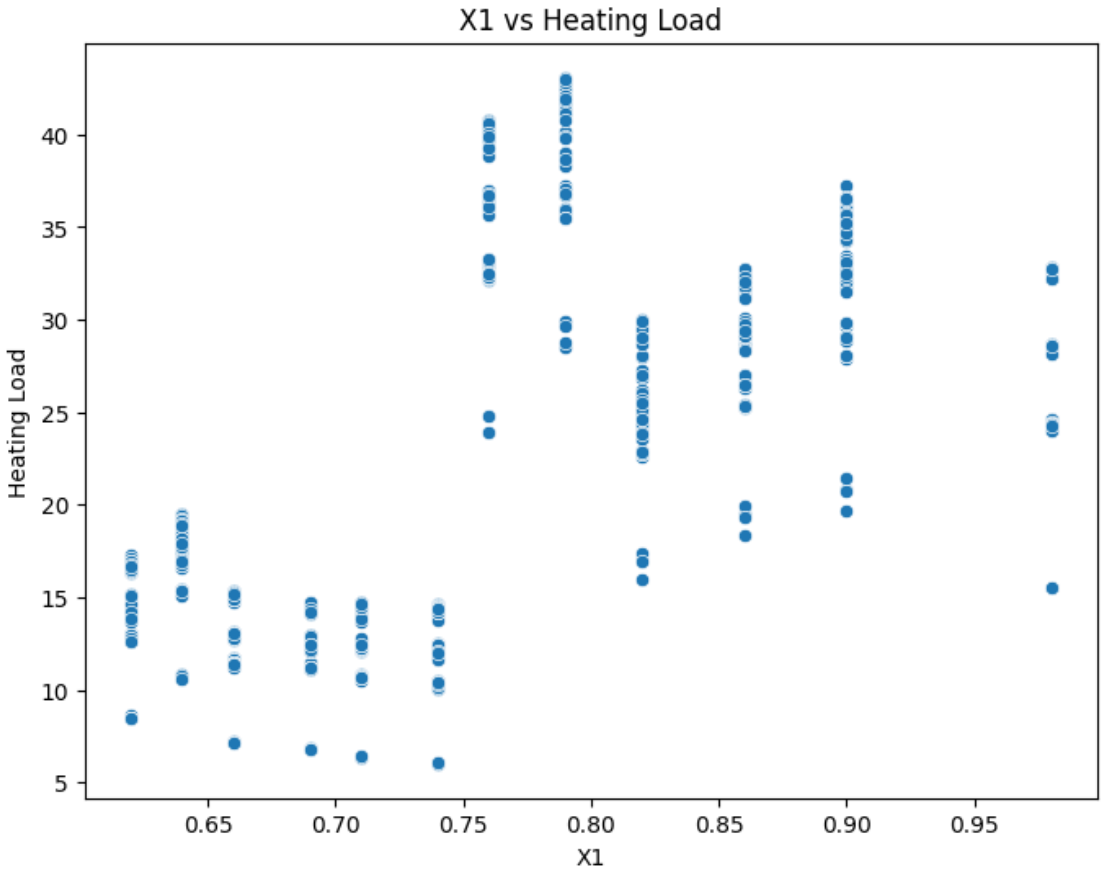
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate the correlation matrix
correlation_matrix = df.corr()

print('Correlation Matrix:')
print(correlation_matrix)

# Create a heatmap of the correlation matrix
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of Building Energy Performance Data')
plt.show()
```

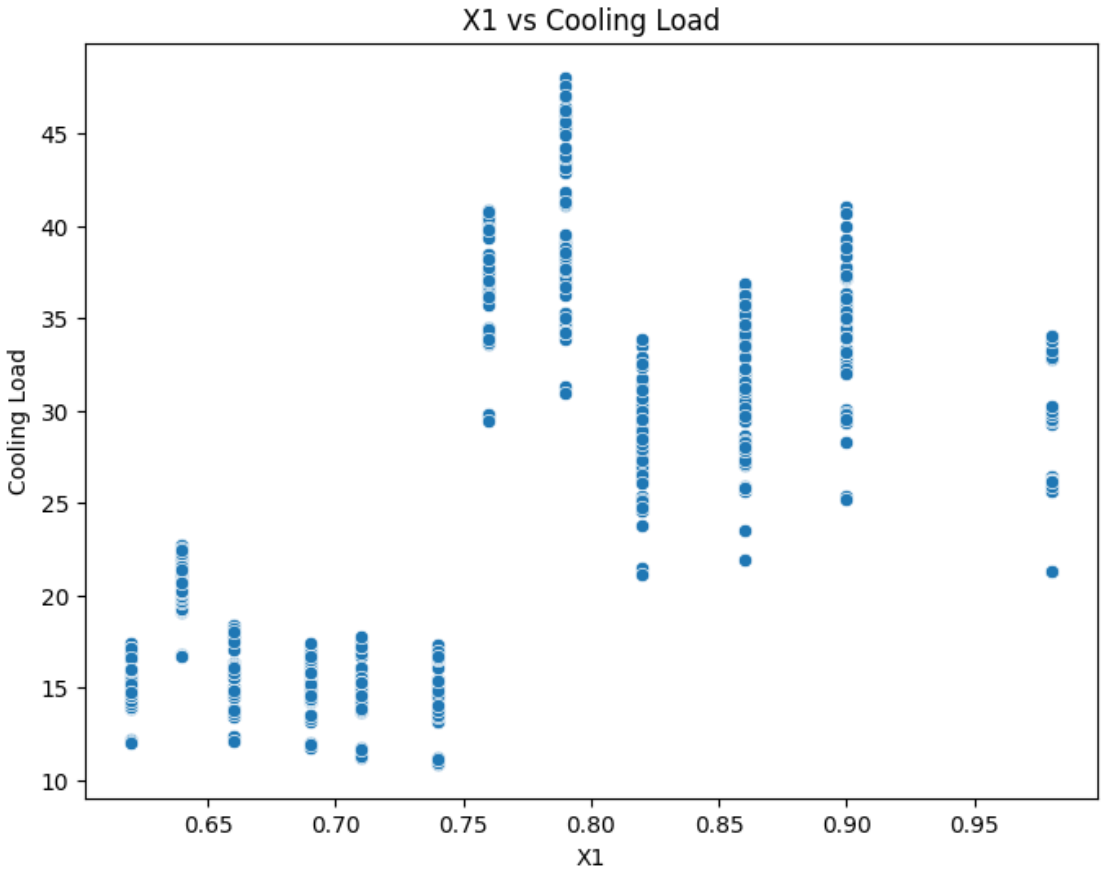
```
Correlation Matrix:
                           X1            X2            X3            X4  \
X1             1.000000e+00 -9.919015e-01 -2.037817e-01 -8.688234e-01
X2            -9.919015e-01  1.000000e+00  1.955016e-01  8.807195e-01
X3            -2.037817e-01  1.955016e-01  1.000000e+00 -2.923165e-01
X4            -8.688234e-01  8.807195e-01 -2.923165e-01  1.000000e+00
X5             8.277473e-01 -8.581477e-01  2.809757e-01 -9.725122e-01
X6             4.678592e-17 -3.459372e-17 -2.429499e-17 -5.830058e-17
X7            -2.960552e-15  3.636925e-15 -8.567455e-17 -1.759011e-15
X8            -7.107006e-16  2.438409e-15  2.067384e-16 -1.078071e-15
Heating Load   6.222719e-01 -6.581199e-01  4.556714e-01 -8.618281e-01
Cooling Load   6.343391e-01 -6.729989e-01  4.271170e-01 -8.625466e-01

                           X5            X6            X7            X8  \
X1             8.277473e-01  4.678592e-17 -2.960552e-15 -7.107006e-16
X2            -8.581477e-01 -3.459372e-17  3.636925e-15  2.438409e-15
X3             2.809757e-01 -2.429499e-17 -8.567455e-17  2.067384e-16
X4            -9.725122e-01 -5.830058e-17 -1.759011e-15 -1.078071e-15
X5             1.000000e+00  4.492205e-17  1.489134e-17 -2.920613e-17
X6             4.492205e-17  1.000000e+00 -9.406007e-16 -2.549352e-16
X7             1.489134e-17 -9.406007e-16  1.000000e+00  2.129642e-01
X8            -2.920613e-17 -2.549352e-16  2.129642e-01  1.000000e+00
Heating Load   8.894305e-01 -2.586763e-03  2.698417e-01  8.736846e-02
Cooling Load   8.957852e-01  1.428960e-02  2.075050e-01  5.052512e-02

               Heating Load  Cooling Load
X1                 0.622272      0.634339
X2                -0.658120     -0.672999
X3                 0.455671      0.427117
X4                -0.861828     -0.862547
X5                 0.889430      0.895785
X6                -0.002587      0.014290
```

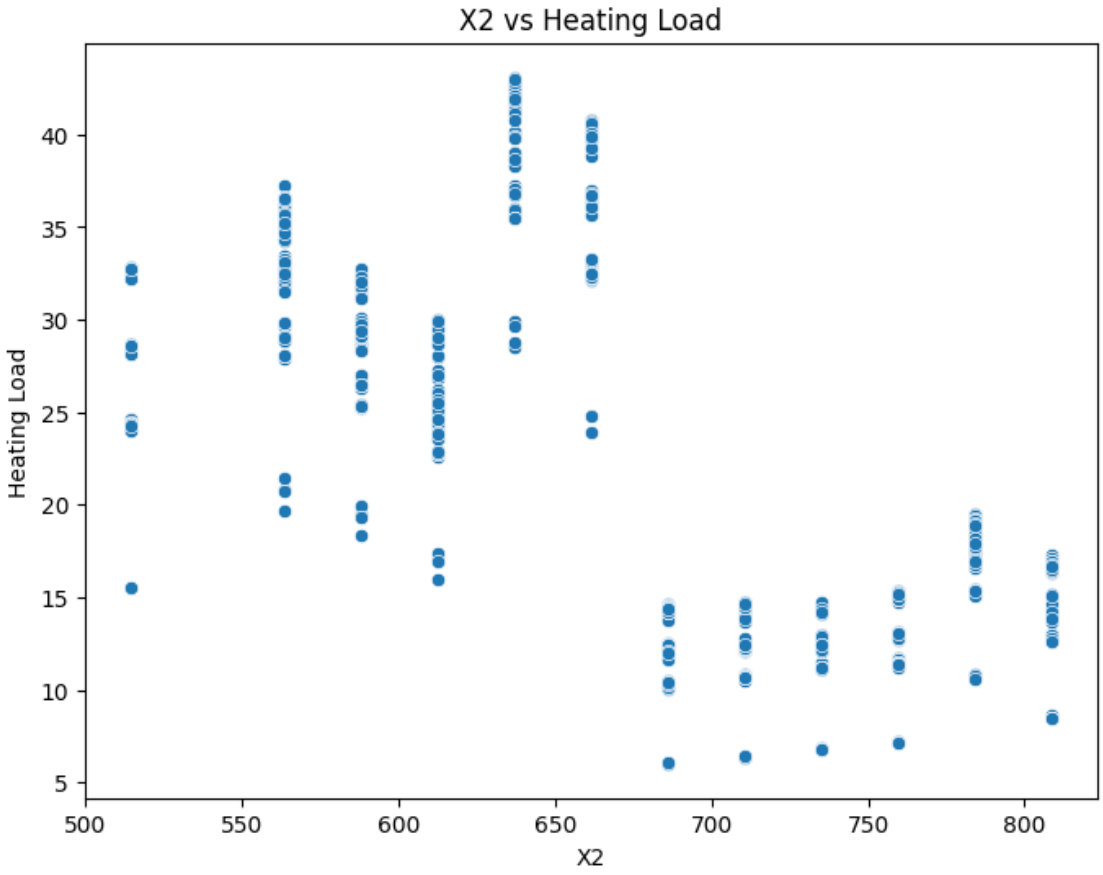
```
X7                 0.269842       0.207505
X8                 0.087368       0.050525
Heating Load       1.000000       0.975862
Cooling Load       0.975862       1.000000
```

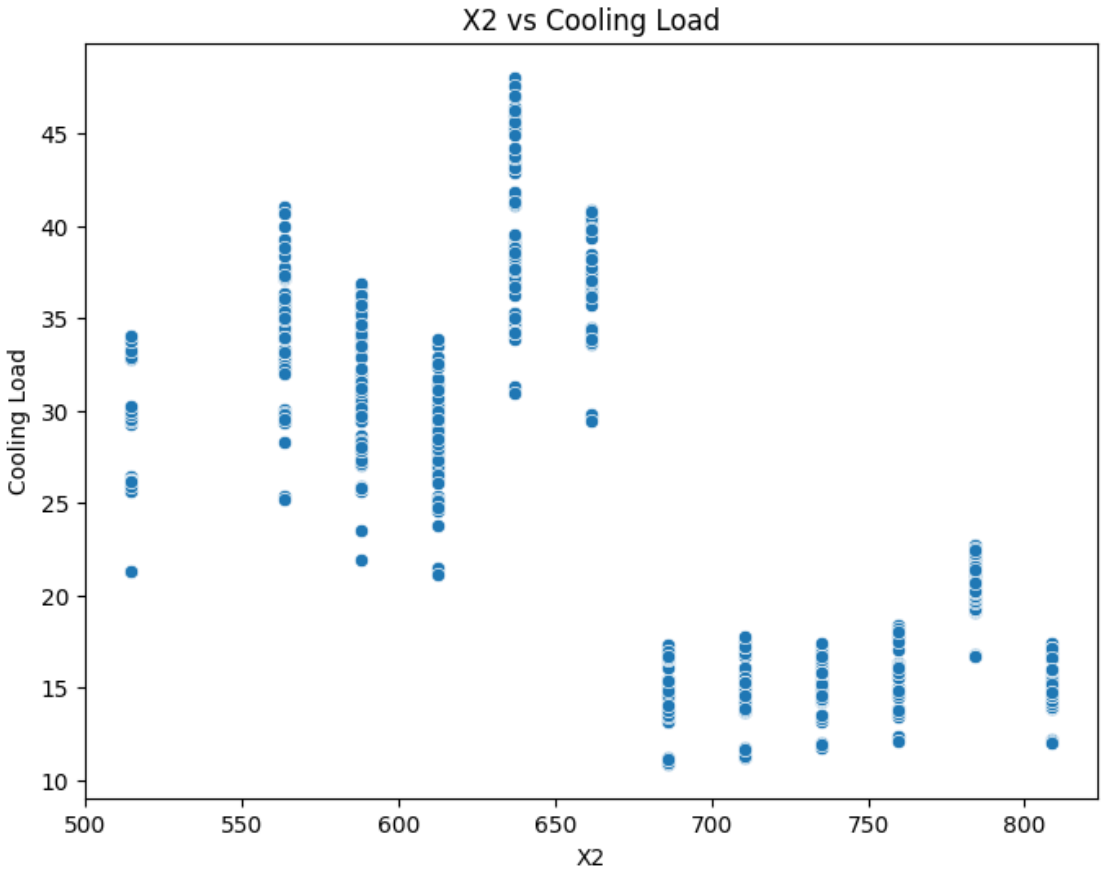


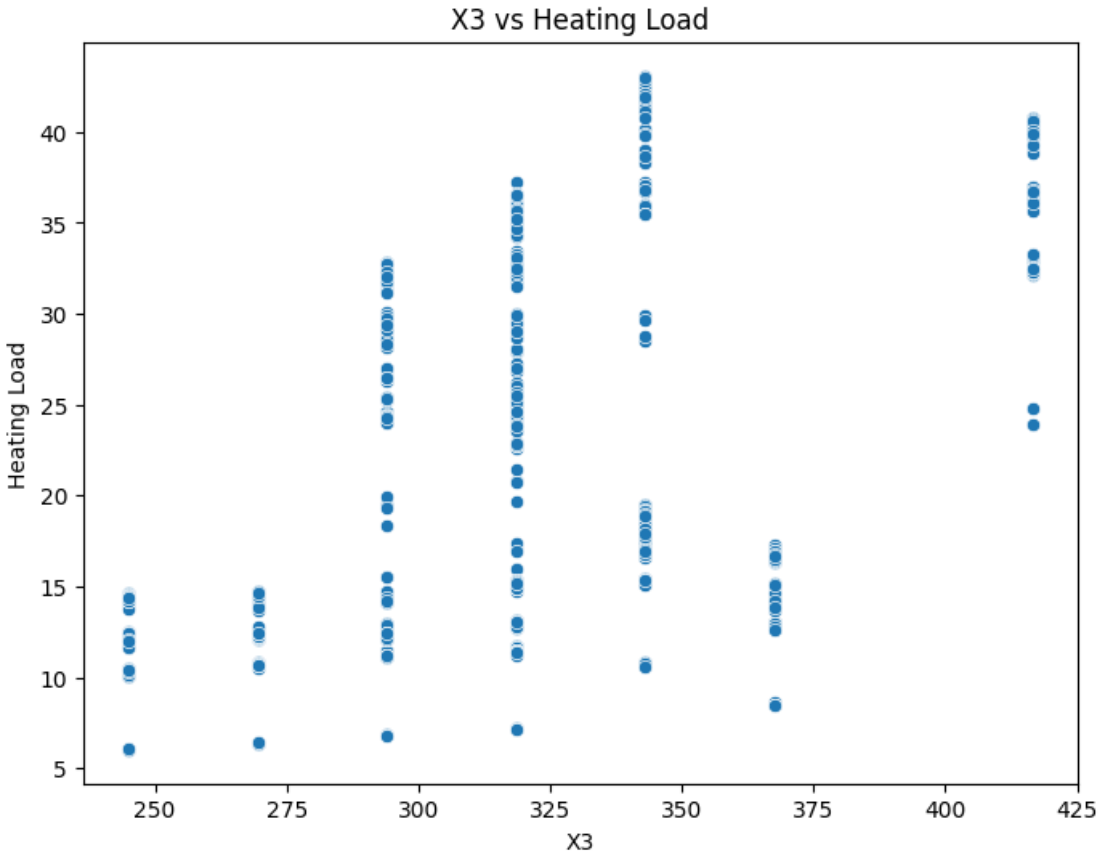

```
independent_features = ['X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8']
target_variables = ['Heating Load', 'Cooling Load']

for feature in independent_features:
    for target in target_variables:
        plt.figure(figsize=(8, 6))
        sns.scatterplot(x=df[feature], y=df[target])
        plt.title(f'{feature} vs {target}')
        plt.xlabel(feature)
        plt.ylabel(target)
        plt.show()
```

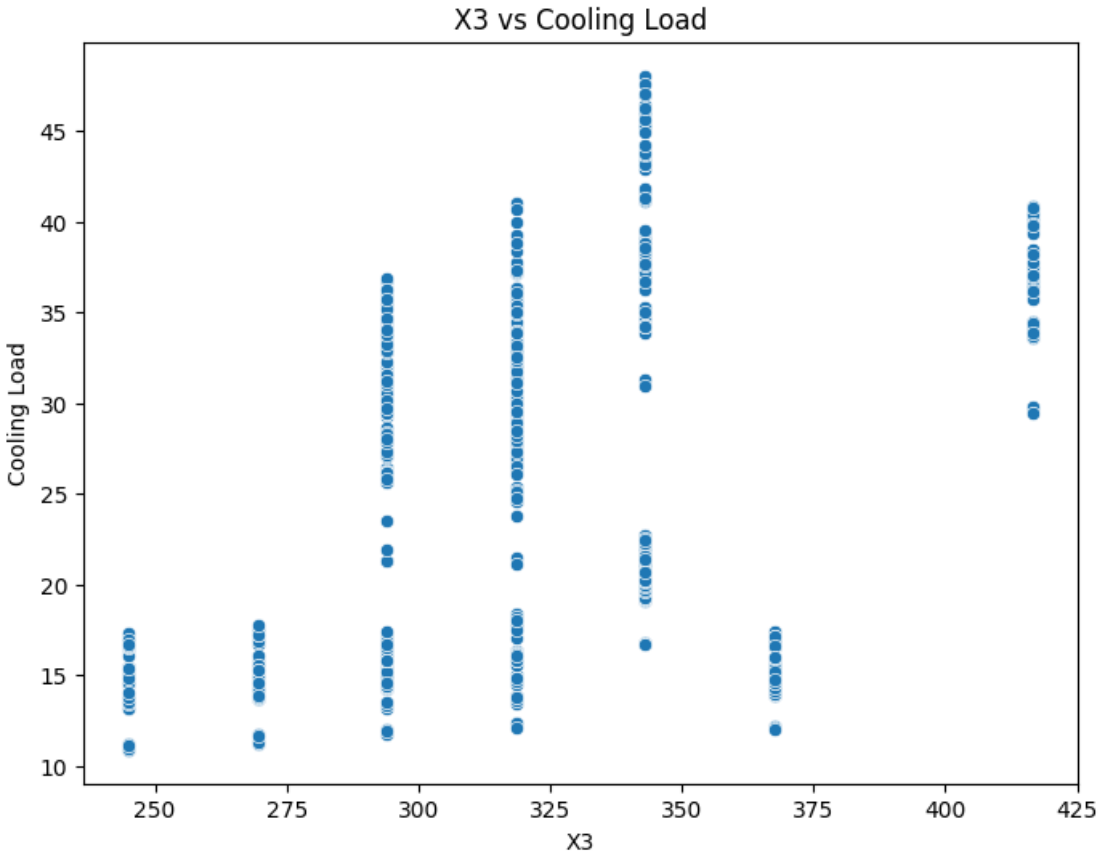

X1 vs Heating Load
Heating Load
X1
40
35
30
25
20
15
10
5
0.65
0.70
0.75
0.80
0.85
0.90
0.95

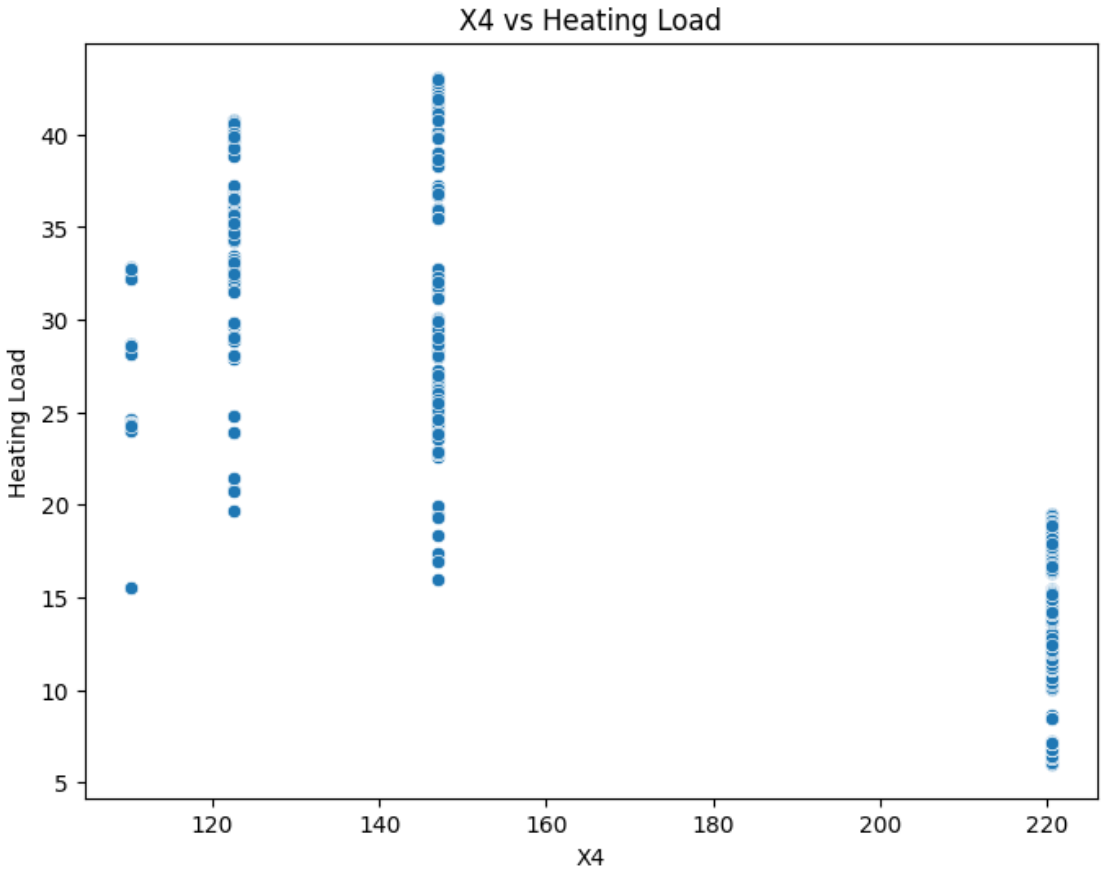

X1 vs Cooling Load
Cooling Load
X1
10
15
20
25
30
35
40
45
0.65
0.70
0.75
0.80
0.85
0.90
0.95

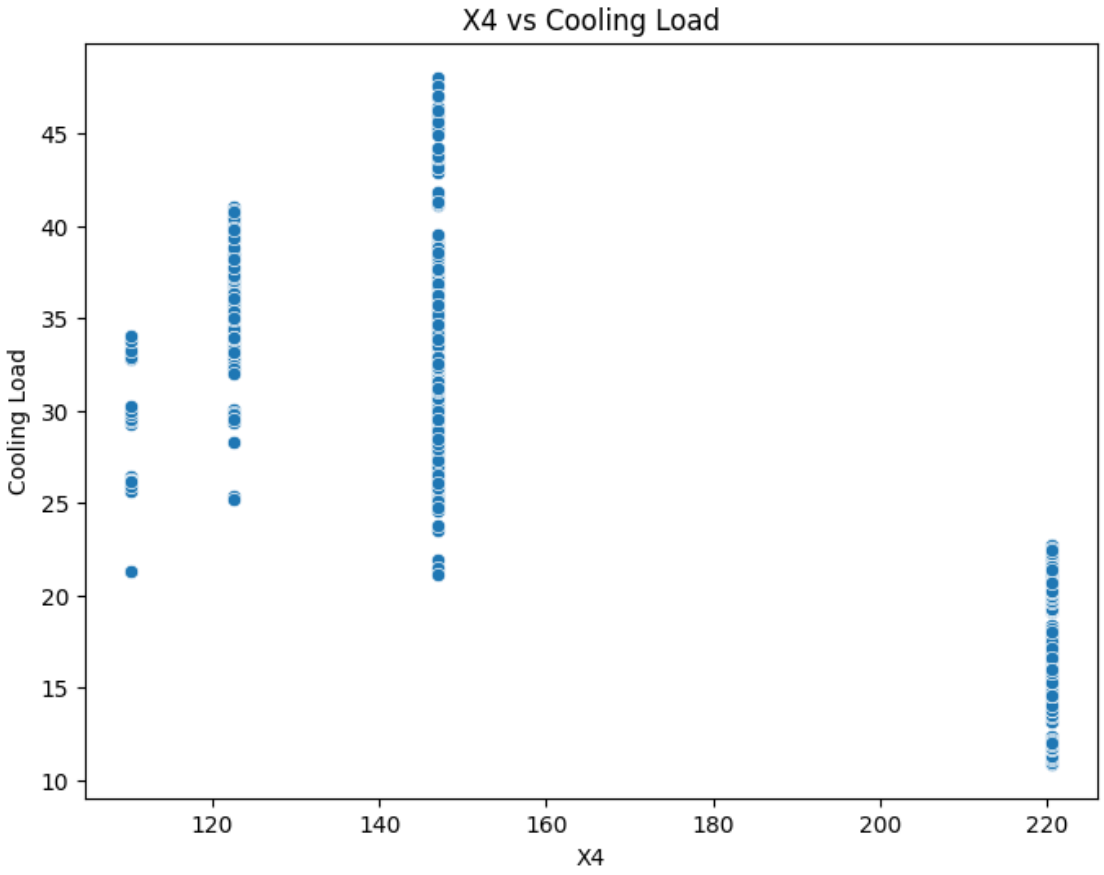

X2 vs Heating Load
Heating Load
X2
40
35
30
25
20
15
10
5
500
550
600
650
700
750
800

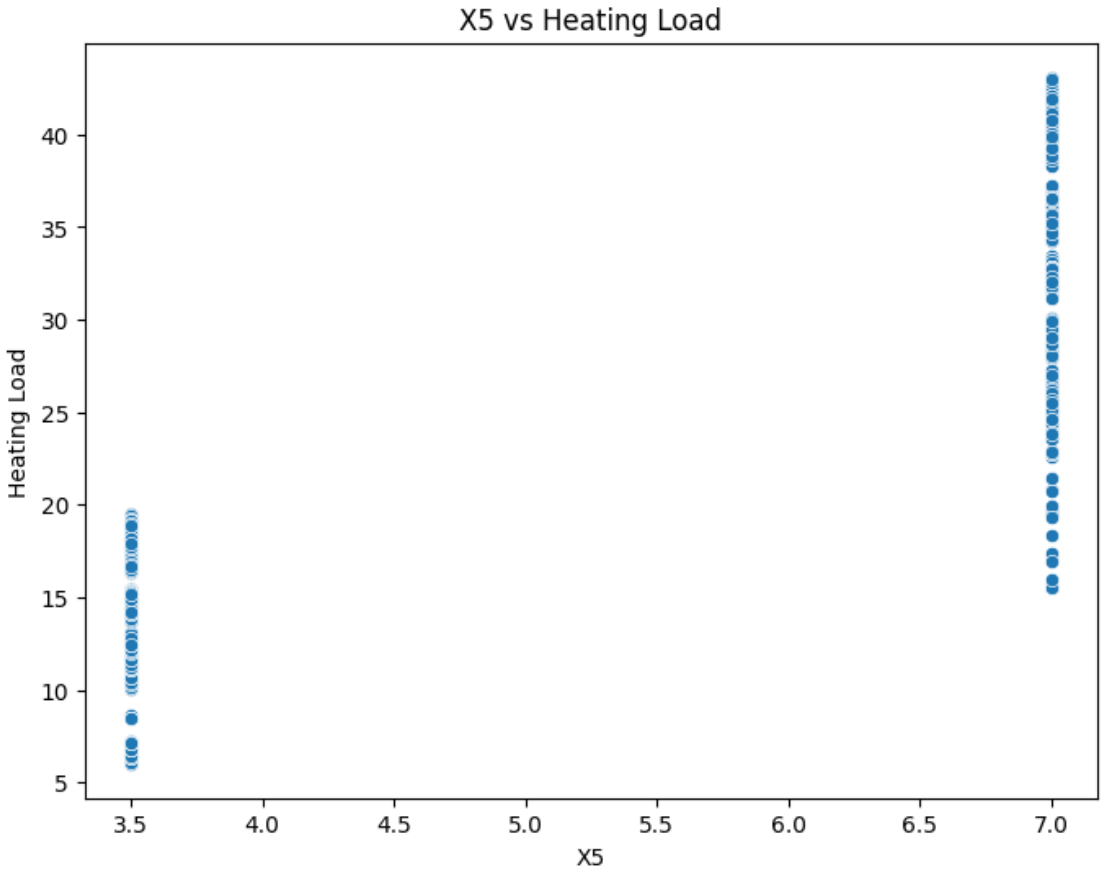

X2 vs Cooling Load
Cooling Load
X2
10
15
20
25
30
35
40
45
500
550
600
650
700
750
800

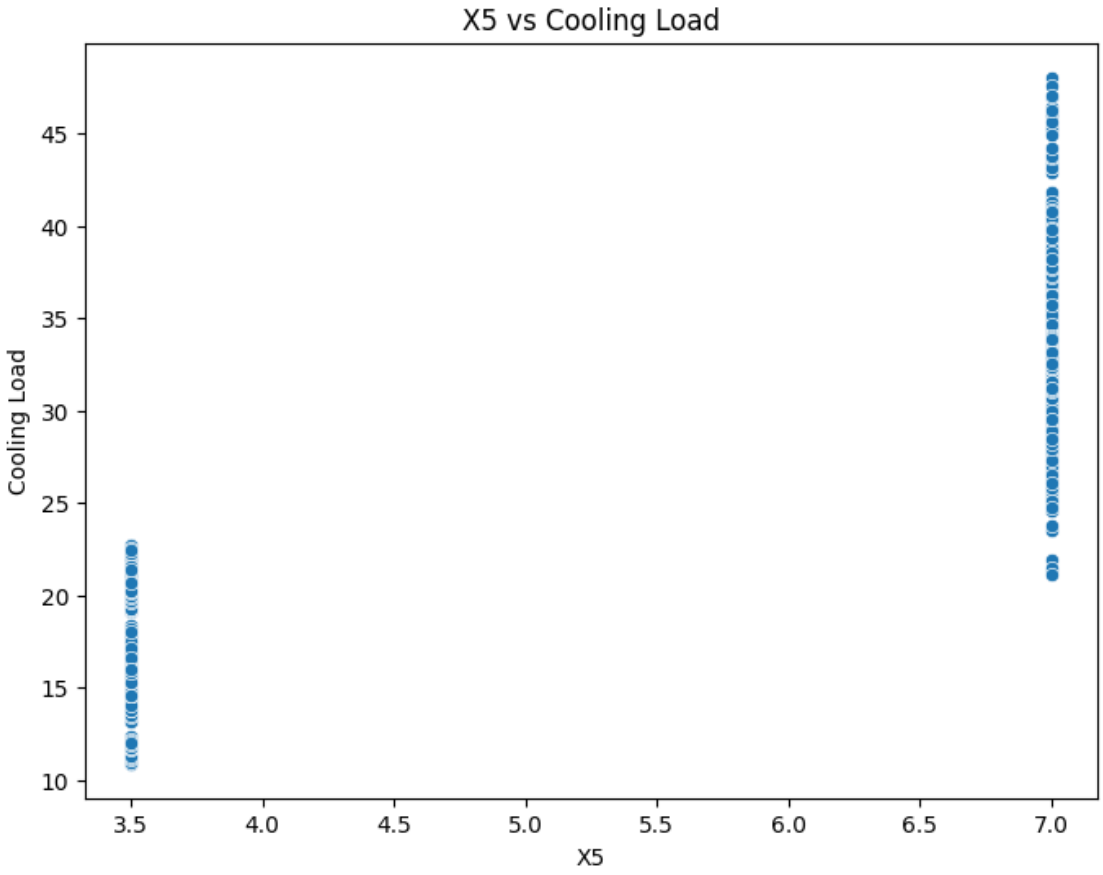

X3 vs Heating Load
Heating Load
X3
5
10
15
20
25
30
35
40
250
275
300
325
350
375
400
425

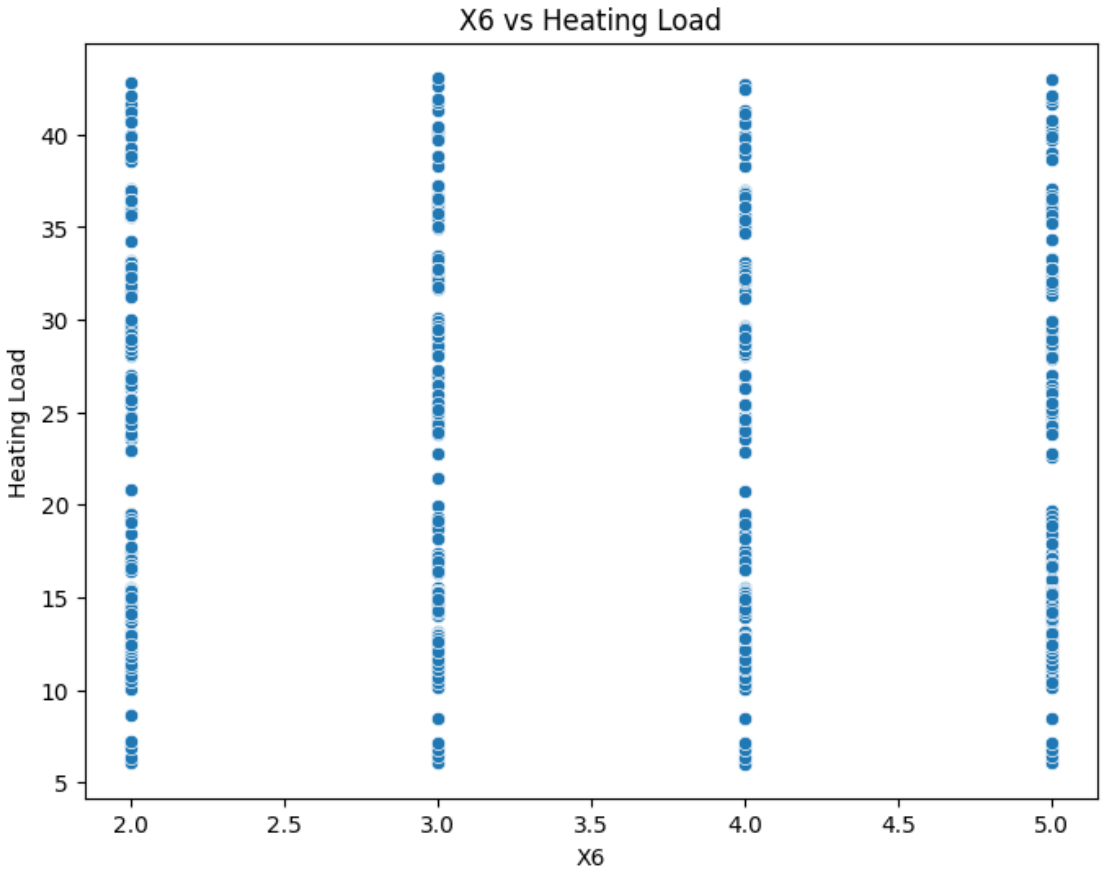

X3 vs Cooling Load
Cooling Load
X3
10
15
20
25
30
35
40
45
250
275
300
325
350
375
400
425

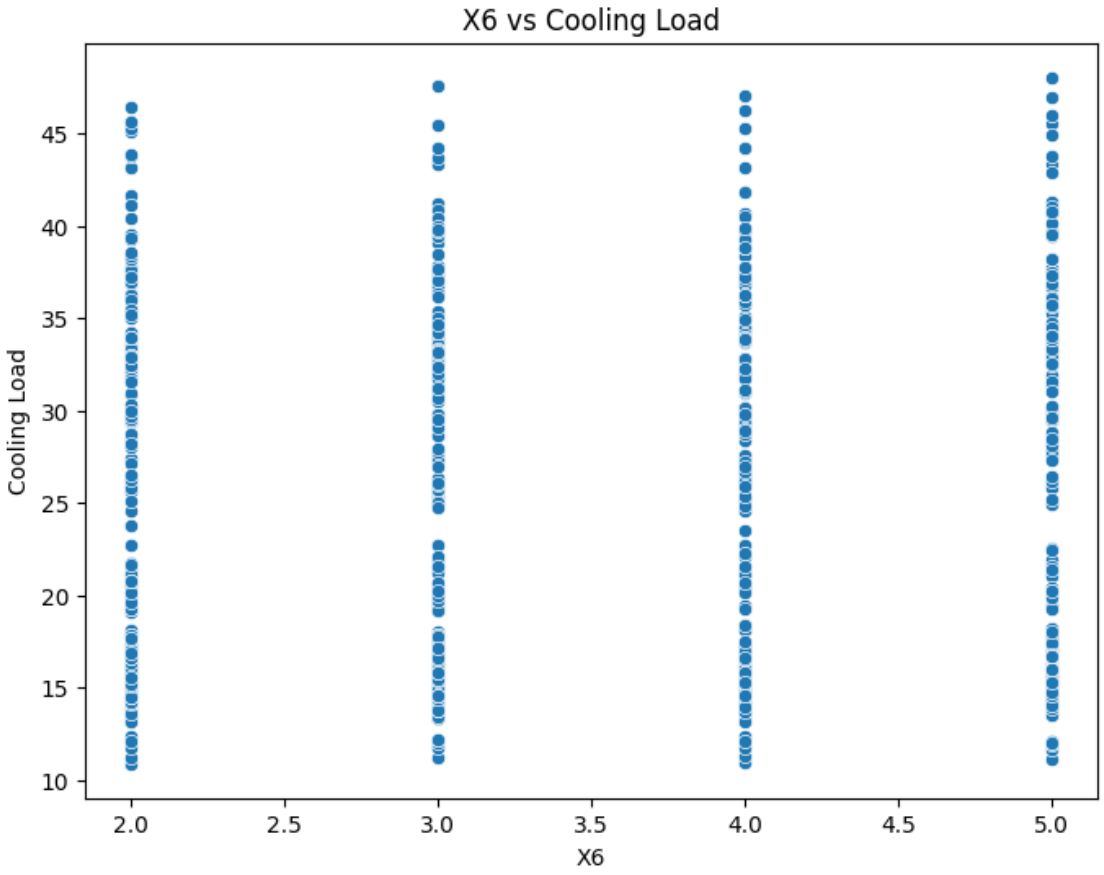

X4 vs Heating Load
Heating Load
X4
40
35
30
25
20
15
10
5
120
140
160
180
200
220

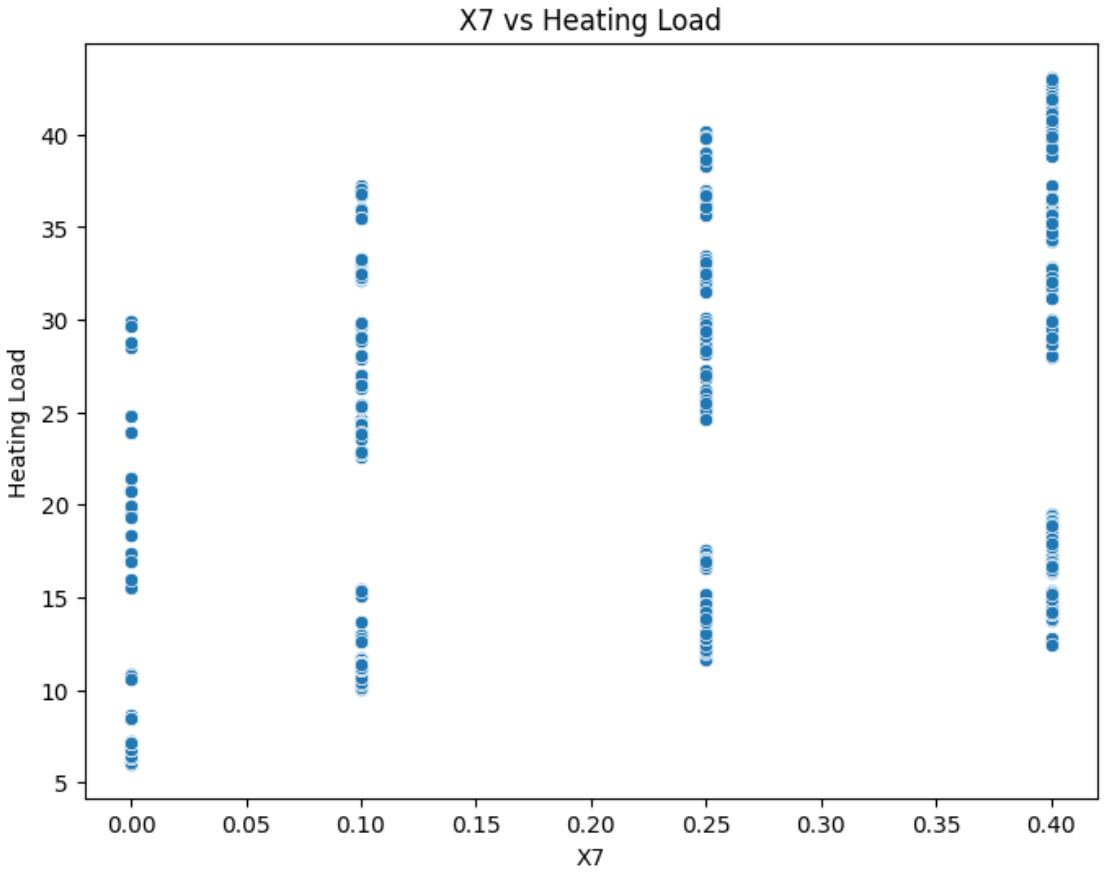

X4 vs Cooling Load
Cooling Load
X4
10
15
20
25
30
35
40
45
120
140
160
180
200
220

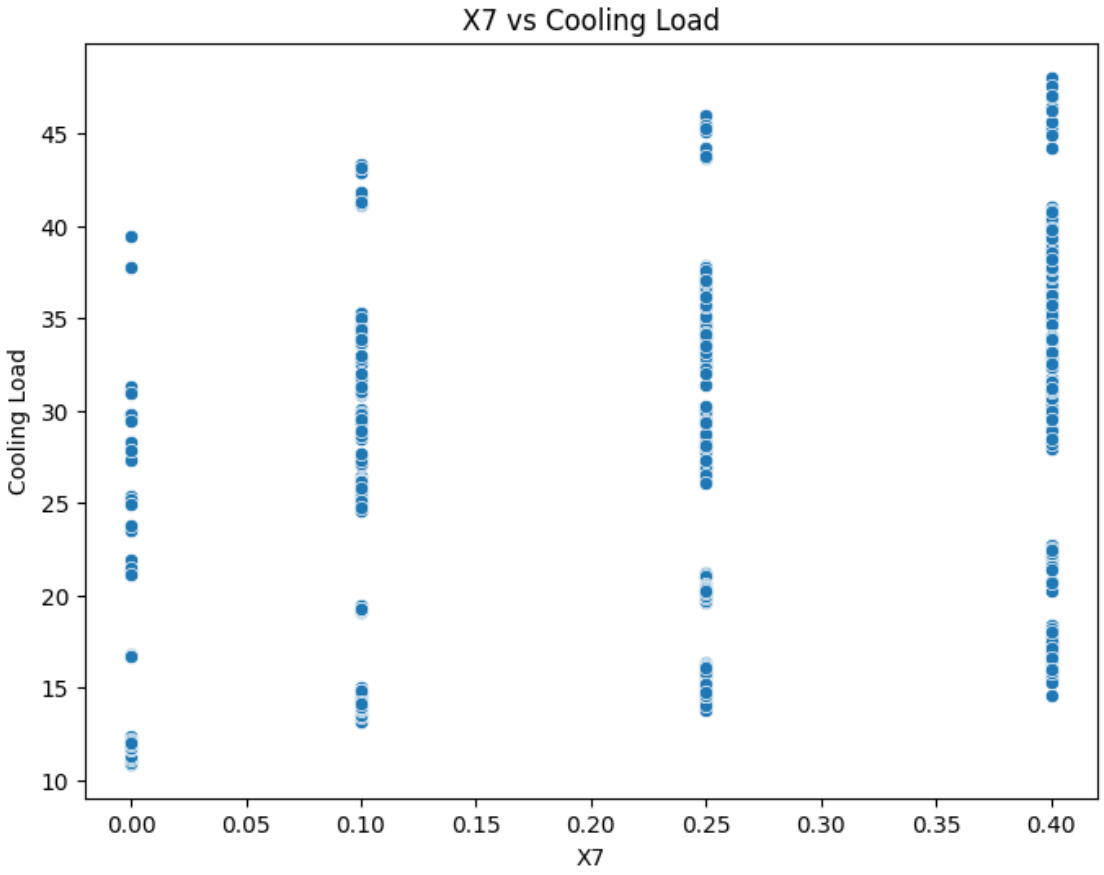

X5 vs Heating Load
Heating Load
X5
40
35
30
25
20
15
10
5
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0

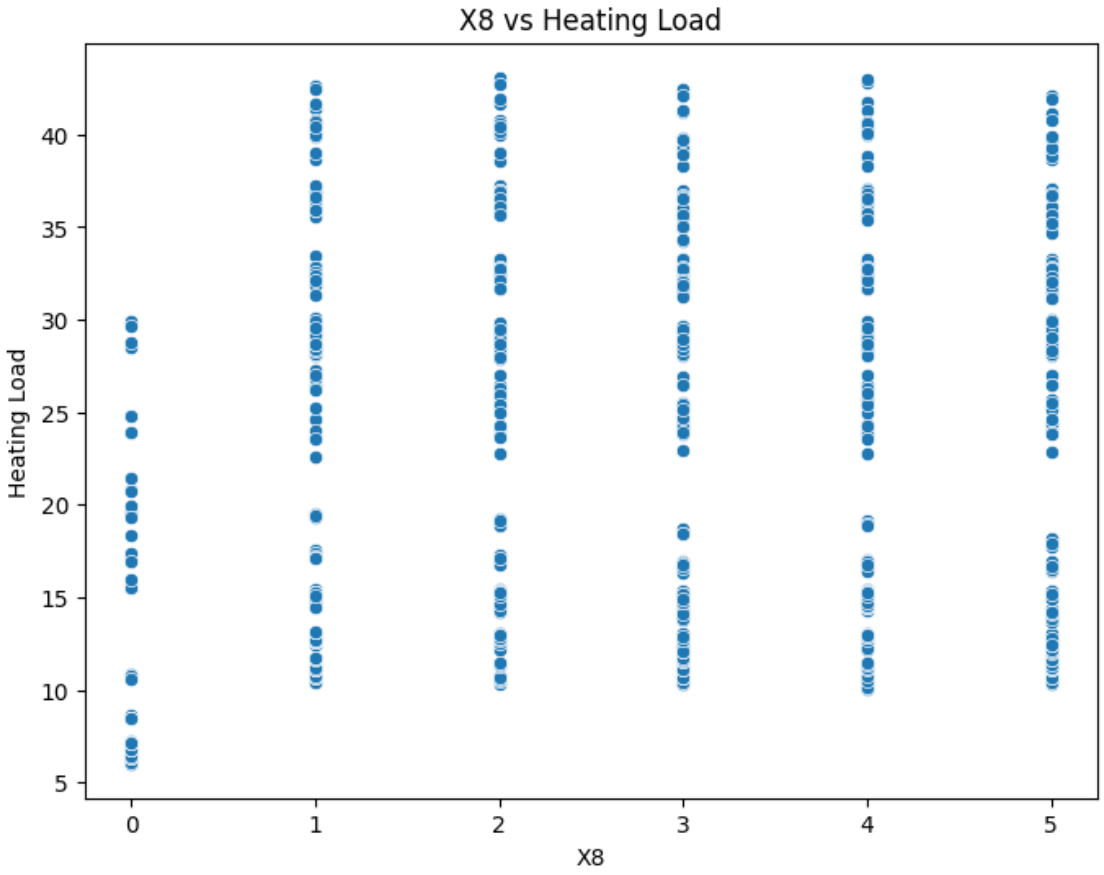

X5 vs Cooling Load
Cooling Load
X5
10
15
20
25
30
35
40
45
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0

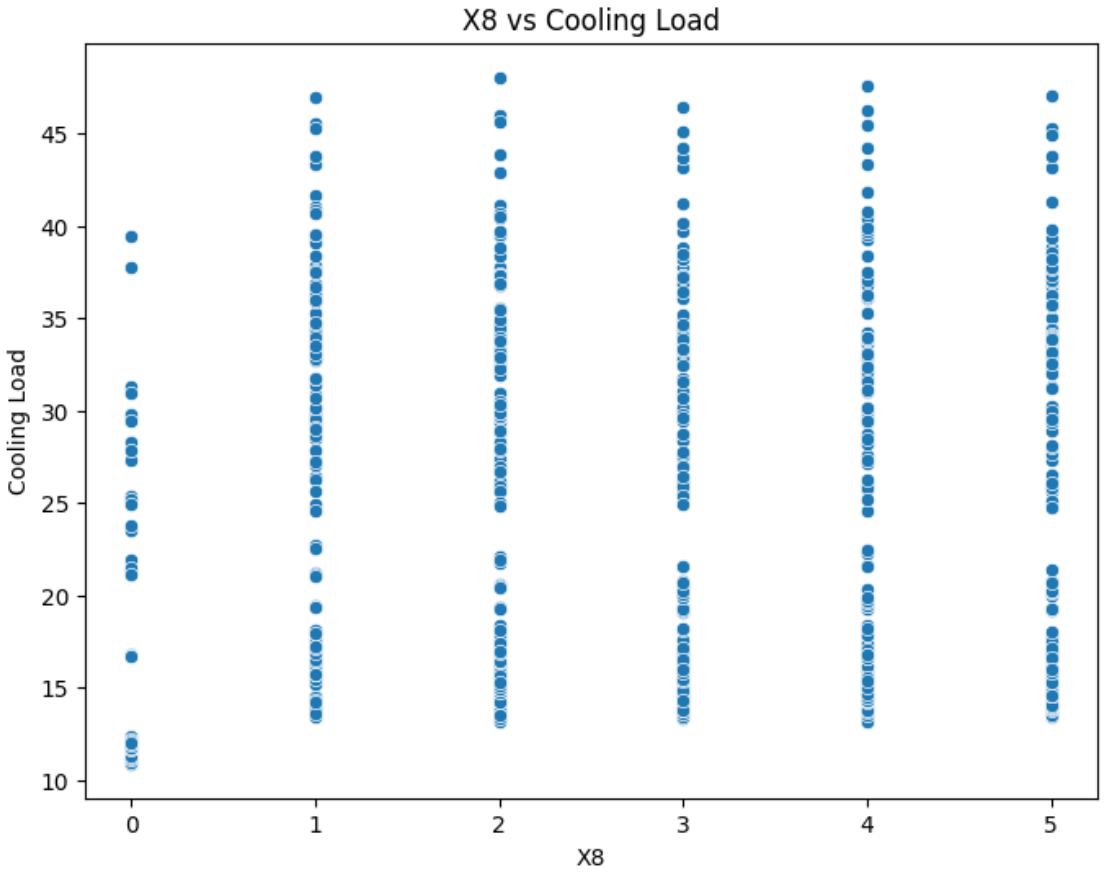

X6 vs Heating Load
Heating Load
X6
40
35
30
25
20
15
10
5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


X6 vs Cooling Load
Cooling Load
X6
10
15
20
25
30
35
40
45
2.0
2.5
3.0
3.5
4.0
4.5
5.0


X7 vs Heating Load
Heating Load
X7
40
35
30
25
20
15
10
5
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40


X7 vs Cooling Load
Cooling Load
X7
10
15
20
25
30
35
40
45
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40


X8 vs Heating Load
Heating Load
X8
5
10
15
20
25
30
35
40
0
1
2
3
4
5



Simple Linear Regression for Heating Load

Subtask:

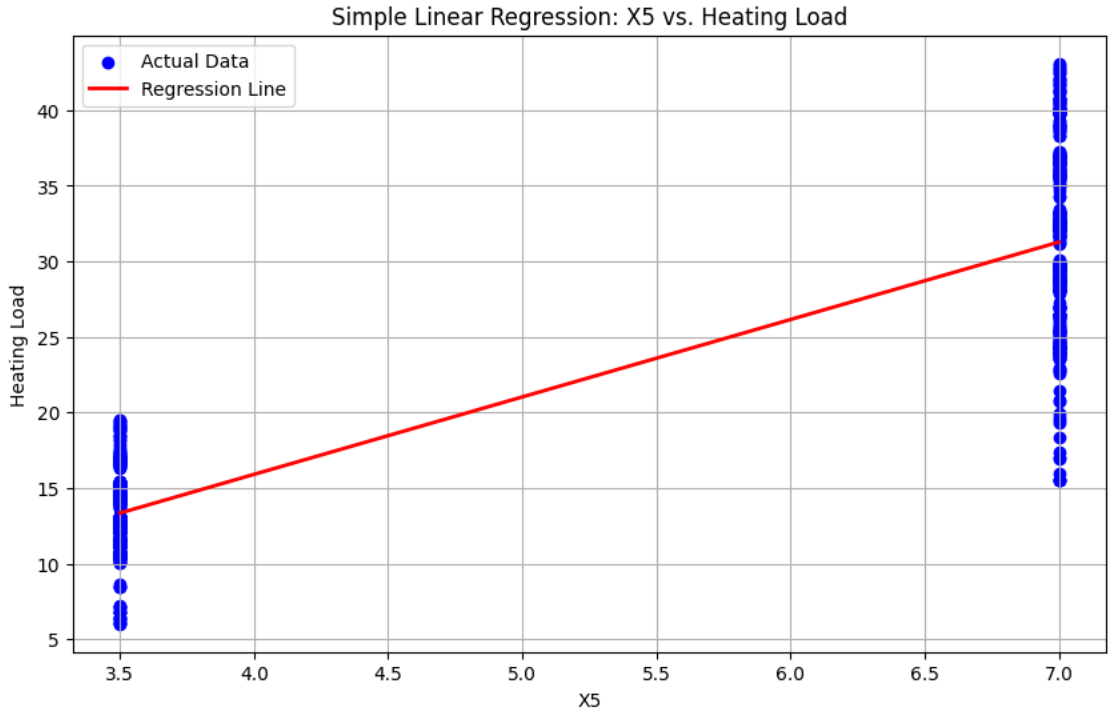
Select the independent feature that shows the strongest linear relationship with 'Heating Load' (based on EDA and correlation). Perform simple linear regression to predict 'Heating Load', visualize the regression line on a scatter plot, and display key regression metrics (e.g., R-squared, RMSE).

Reasoning: Based on the previous correlation analysis, 'X5' was identified as having the strongest positive linear relationship with 'Heating Load'. This step will perform simple linear regression using 'X5' to predict 'Heating Load', evaluate the model's performance, and visualize the regression line.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import numpy as np

# Select the independent feature (X5) and the target variable (Heating Load)
X = df[['X5']]
Y = df['Heating Load']
```

```python
# Instantiate and fit the Linear Regression model
model = LinearRegression()
model.fit(X, Y)

# Predict 'Heating Load' values
Y_pred = model.predict(X)

# Calculate and print regression metrics
r_squared = r2_score(Y, Y_pred)
mse = mean_squared_error(Y, Y_pred)
rmse = np.sqrt(mse)

print(f"R-squared: {r_squared:.4f}")
print(f"Mean Squared Error (MSE): {mse:.4f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.4f}")

# Visualize the regression line on a scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(X, Y, color='blue', label='Actual Data')
plt.plot(X, Y_pred, color='red', linewidth=2, label='Regression Line')
plt.title('Simple Linear Regression: X5 vs. Heating Load')
plt.xlabel('X5')
plt.ylabel('Heating Load')
plt.legend()
plt.grid(True)
plt.show()
```

```
R-squared: 0.7911
Mean Squared Error (MSE): 21.2422
Root Mean Squared Error (RMSE): 4.6089
```



Simple Linear Regression for Cooling Load

Subtask:

Select the independent feature that shows the strongest linear relationship with 'Cooling Load' (based on EDA and correlation). Perform simple linear regression to predict 'Cooling Load', visualize the regression line on a scatter plot, and display key regression metrics (e.g., R-squared, RMSE).

Reasoning:

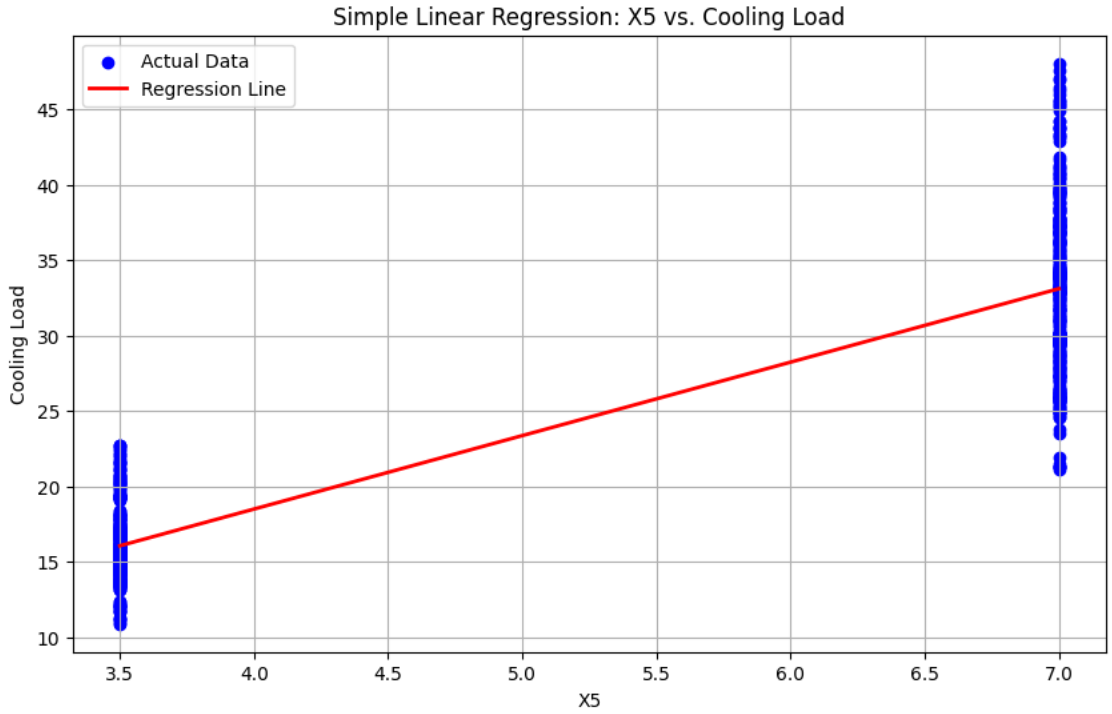
Based on the previous correlation analysis, 'X5' was identified as having the strongest positive linear relationship with 'Cooling Load'. This step will perform simple linear regression using 'X5' to predict 'Cooling Load', evaluate the model's performance, and visualize the regression line.

Reasoning: This step performs simple linear regression to predict 'Cooling Load' using 'X5' as the independent variable, as identified from the correlation analysis. It includes model fitting, prediction, metric calculation (R-squared, MSE, RMSE), and visualization of the regression line on a scatter plot.

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import numpy as np

# Select the independent feature (X5) and the target variable (Cooling Load)
X_cooling = df[['X5']]
```

```
Y_cooling = df['Cooling Load']

# Instantiate and fit the Linear Regression model
model_cooling = LinearRegression()
model_cooling.fit(X_cooling, Y_cooling)

# Predict 'Cooling Load' values
Y_pred_cooling = model_cooling.predict(X_cooling)

# Calculate and print regression metrics
r_squared_cooling = r2_score(Y_cooling, Y_pred_cooling)
mse_cooling = mean_squared_error(Y_cooling, Y_pred_cooling)
rmse_cooling = np.sqrt(mse_cooling)

print(f"R-squared for Cooling Load: {r_squared_cooling:.4f}")
print(f"Mean Squared Error (MSE) for Cooling Load: {mse_cooling:.4f}")
print(f"Root Mean Squared Error (RMSE) for Cooling Load: {rmse_cooling:.4f}")

# Visualize the regression line on a scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(X_cooling, Y_cooling, color='blue', label='Actual Data')
plt.plot(X_cooling, Y_pred_cooling, color='red', linewidth=2,
label='Regression Line')
plt.title('Simple Linear Regression: X5 vs. Cooling Load')
plt.xlabel('X5')
plt.ylabel('Cooling Load')
plt.legend()
plt.grid(True)
plt.show()
```

```
R-squared for Cooling Load: 0.8024
Mean Squared Error (MSE) for Cooling Load: 17.8573
Root Mean Squared Error (RMSE) for Cooling Load: 4.2258
```



Summary:

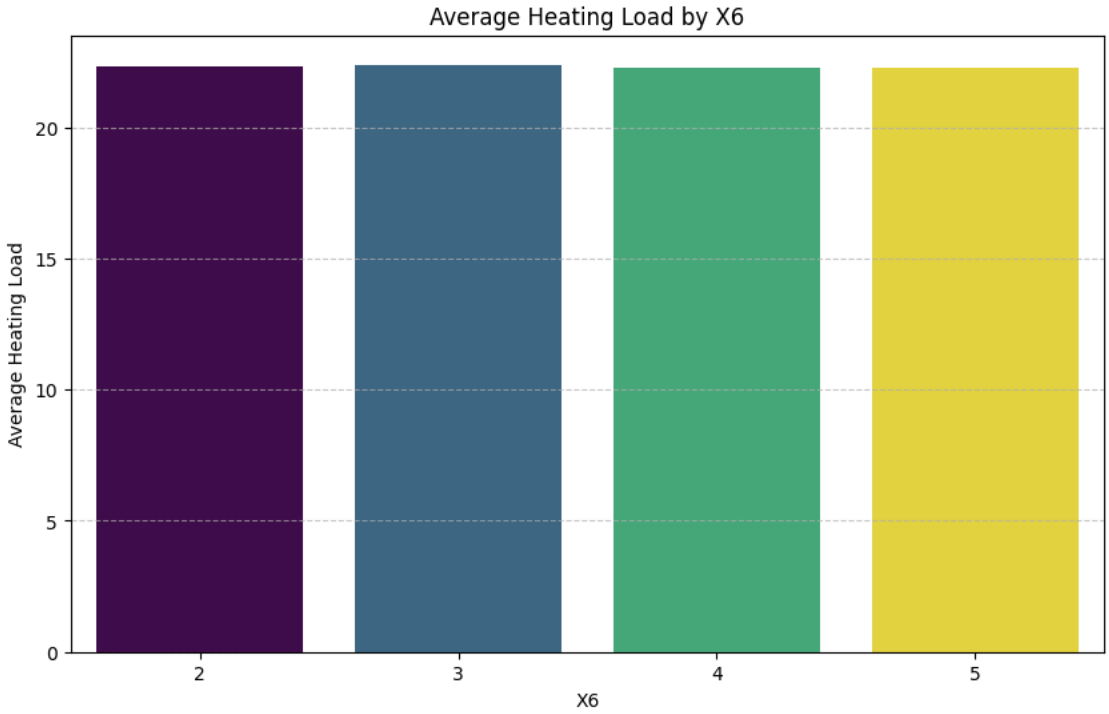
Data Analysis Key Findings

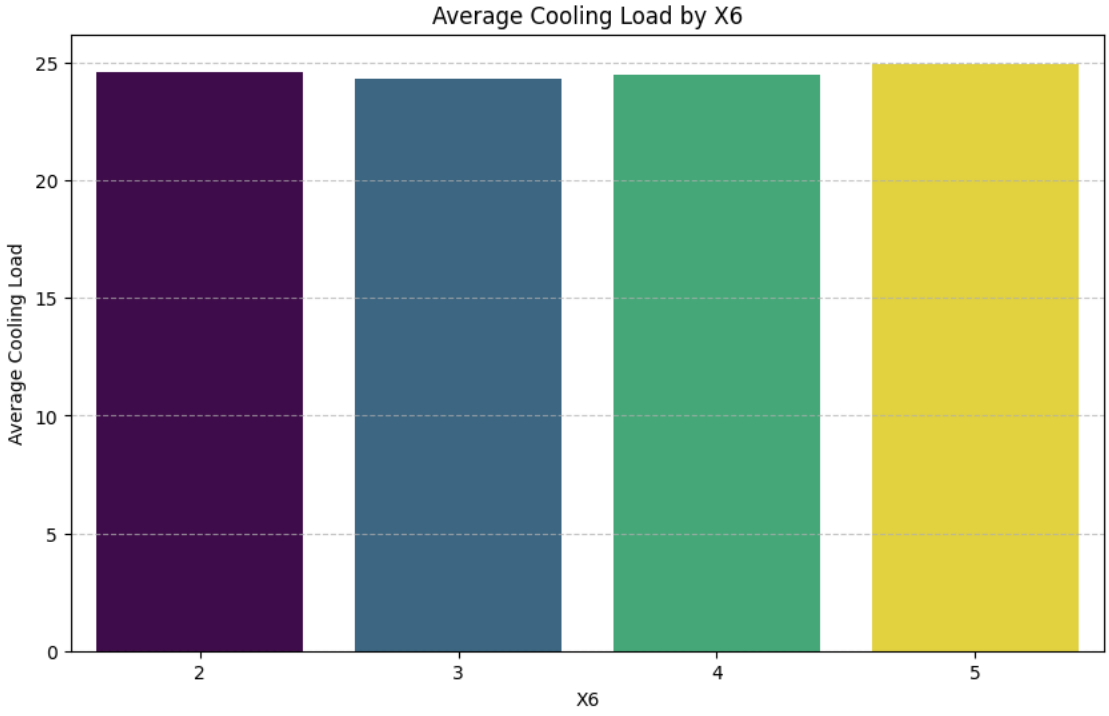
- **Most Influential Features:**
  - For **Heating Load**, X5 (Relative Compactness) showed the strongest positive correlation (0.889), and X4 (Wall Area) exhibited the strongest negative correlation (-0.862). X2 (Surface Area) also had a strong negative correlation (-0.658).
  - For **Cooling Load**, X5 (Relative Compactness) displayed the strongest positive correlation (0.896), and X4 (Wall Area) showed the strongest negative correlation (-0.863). X2 (Surface Area) also had a strong negative correlation (-0.673).
  - X6 (Overall Height), X7 (Orientation), and X8 (Glazing Distribution) showed much weaker correlations with both heating and cooling loads.
- **Simple Linear Regression Performance for Heating Load (using X5):**
  - The model achieved an R-squared value of 0.7911, indicating that approximately 79.11% of the variance in 'Heating Load' can be explained by X5.
  - The Root Mean Squared Error (RMSE) for 'Heating Load' was 4.6089.
- **Simple Linear Regression Performance for Cooling Load (using X5):**
  - The model achieved an R-squared value of 0.8024, indicating that approximately 80.24% of the variance in 'Cooling Load' can be explained by X5.
  - The Root Mean Squared Error (RMSE) for 'Cooling Load' was 4.2258.

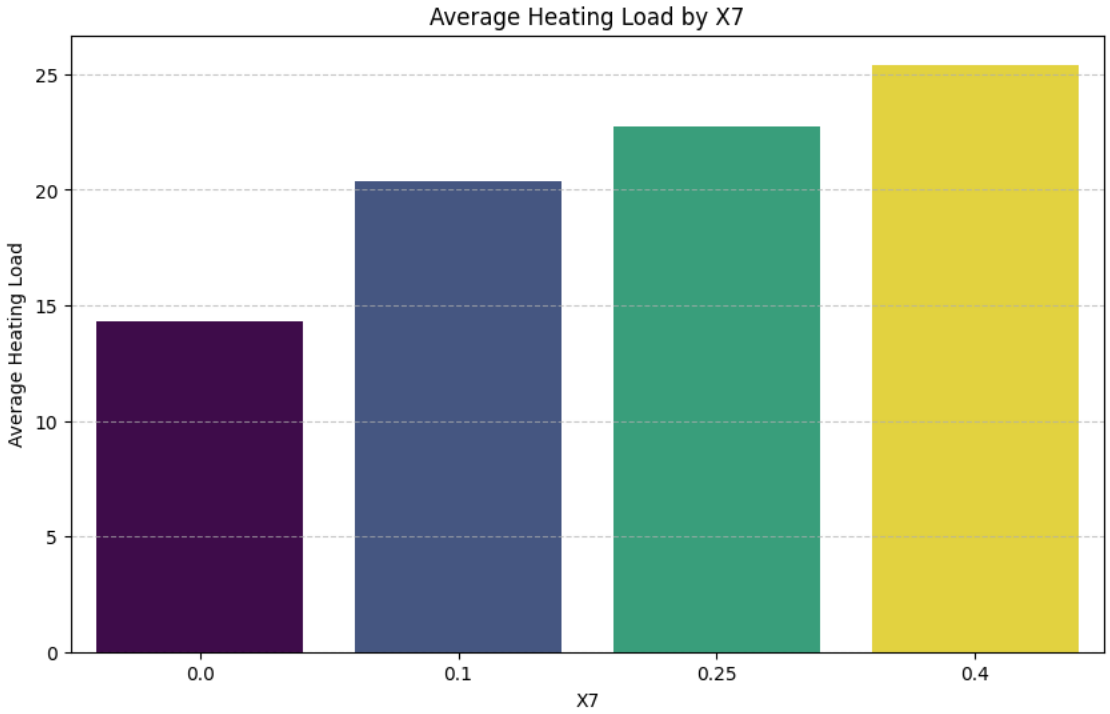
Insights or Next Steps

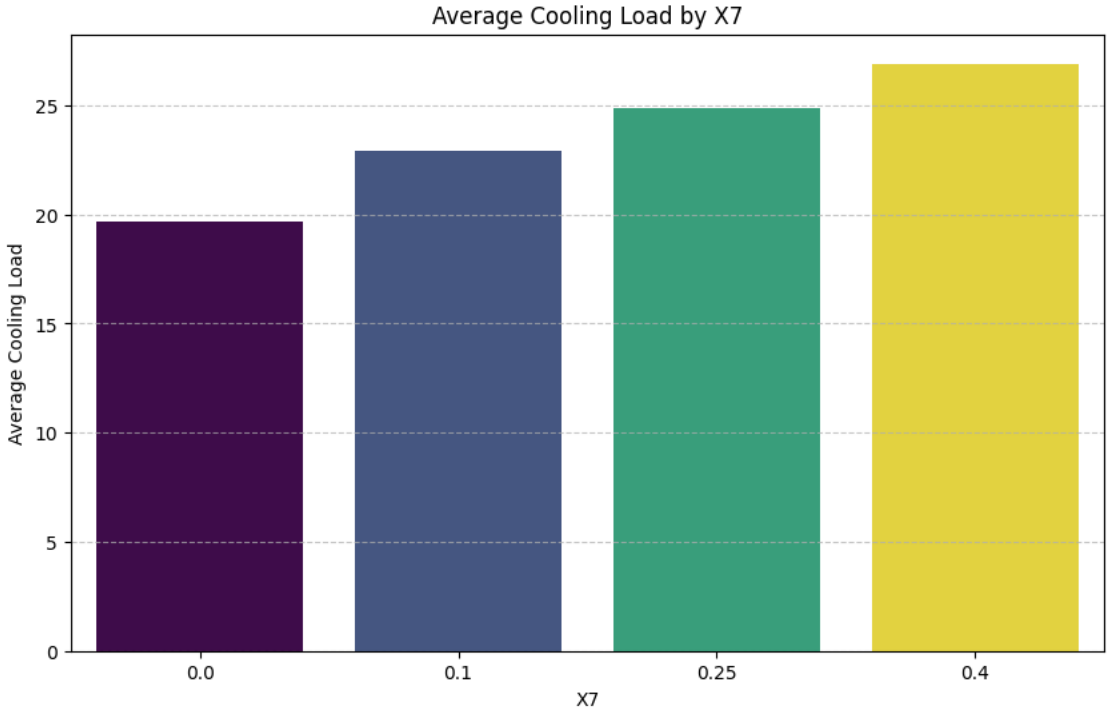
- **Multivariate Analysis:** While X5 is highly influential, consider incorporating other strongly correlated features like X4 and X2 into a multiple linear regression model to potentially improve predictive accuracy for both 'Heating Load' and 'Cooling Load'.
- **Feature Engineering/Non-linear Models:** Investigate potential non-linear relationships or interactions between features, especially for those with weaker linear correlations, by exploring polynomial regression or other non-linear modeling techniques.

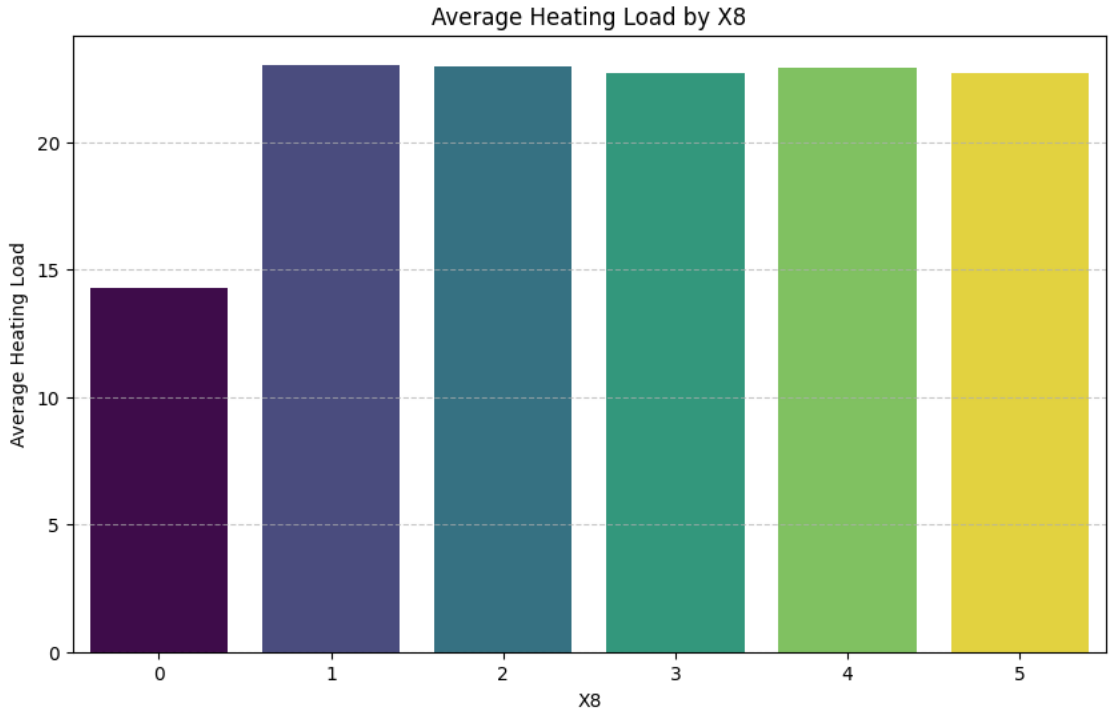
```
import matplotlib.pyplot as plt
import seaborn as sns

categorical_features = ['X6', 'X7', 'X8']
target_variables = ['Heating Load', 'Cooling Load']

for feature in categorical_features:
    for target in target_variables:
        plt.figure(figsize=(10, 6))
        # Calculate the mean of the target variable for each category of the
feature
        avg_loads = df.groupby(feature)[target].mean().reset_index()
        sns.barplot(x=feature, y=target, data=avg_loads, palette='viridis',
hue=feature, legend=False)
        plt.title(f'Average {target} by {feature}')
        plt.xlabel(feature)
        plt.ylabel(f'Average {target}')
        plt.grid(axis='y', linestyle='--', alpha=0.7)
        plt.show()
```

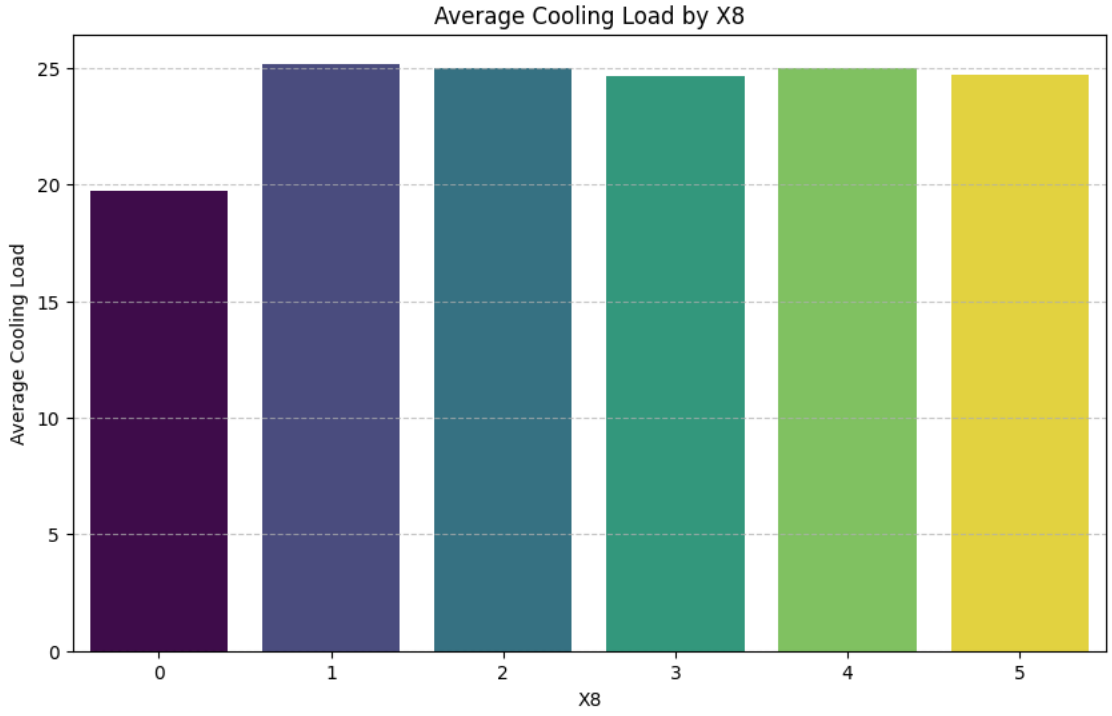

Average Heating Load by X6
Average Heating Load
0
5
10
15
20
2
3
4
5
X6


Average Cooling Load by X6
Average Cooling Load
0
5
10
15
20
25
2
3
4
5
X6


Average Heating Load by X7
Average Heating Load
0
5
10
15
20
25
0.0
0.1
0.25
0.4
X7


Average Cooling Load by X7
Average Cooling Load
0
5
10
15
20
25
0.0
0.1
0.25
0.4
X7





Graph Summary and Conclusion

Here is a summary of all the graphs generated, along with what each represents:

1. **Correlation Matrix Heatmap:** This heatmap (Figure 1 in `c3756396`) illustrates the correlation coefficients between all features (X1-X8) and target variables ('Heating Load', 'Cooling Load') in the dataset.
1. **Scatter Plots of Independent Features vs. Heating Load:** These 8 scatter plots (Figures 1-8 in `86cab082`) visually represent the relationship between each independent feature (X1-X8) and the 'Heating Load', helping to assess potential linear relationships.
2. **Scatter Plots of Independent Features vs. Cooling Load:** These 8 scatter plots (Figures 9-16 in `86cab082`) visually represent the relationship between each independent feature (X1-X8) and the 'Cooling Load', helping to assess potential linear relationships.
3. **Simple Linear Regression: X5 vs. Heating Load:** This scatter plot with a regression line (Figure 1 in `273ca758`) shows the linear relationship between 'X5' (Relative Compactness) and 'Heating Load', along with the model's fit and performance metrics (R-squared: 0.7911, RMSE: 4.6089).
4. **Simple Linear Regression: X5 vs. Cooling Load:** This scatter plot with a regression line (Figure 1 in `af336af1`) shows the linear relationship between 'X5' (Relative Compactness) and 'Cooling Load', along with the model's fit and performance metrics (R-squared: 0.8024, RMSE: 4.2258).
5. **Average Heating Load by X6 (Overall Height):** This bar plot (Figure 1 in `a1bb8316`) displays the average 'Heating Load' for different categories of 'X6' (Overall Height), revealing its impact on heating requirements.
6. **Average Cooling Load by X6 (Overall Height):** This bar plot (Figure 2 in `a1bb8316`) displays the average 'Cooling Load' for different categories of 'X6' (Overall Height), revealing its impact on cooling requirements.
7. **Average Heating Load by X7 (Orientation):** This bar plot (Figure 3 in `a1bb8316`) displays the average 'Heating Load' for different categories of 'X7' (Orientation), showing how building orientation affects heating.
8. **Average Cooling Load by X7 (Orientation):** This bar plot (Figure 4 in `a1bb8316`) displays the average 'Cooling Load' for different categories of 'X7' (Orientation), showing how building orientation affects cooling.
9. **Average Heating Load by X8 (Glazing Distribution):** This bar plot (Figure 5 in `a1bb8316`) displays the average 'Heating Load' for different categories of 'X8' (Glazing Distribution), illustrating its effect on heating.
10. **Average Cooling Load by X8 (Glazing Distribution):** This bar plot (Figure 6 in `a1bb8316`) displays the average 'Cooling Load' for different categories of 'X8' (Glazing Distribution), illustrating its effect on cooling.

Conclusion:

From the analysis, we can conclude the following:

- **Relative Compactness (X5)** is the most influential feature for both heating and cooling loads, exhibiting a strong positive linear relationship. This is evident from

the high correlation coefficients and the good performance of the simple linear regression models using X5.

- **Wall Area (X4)** and **Surface Area (X2)** also show strong negative correlations with both loads, indicating their significance.
- **Overall Height (X6), Orientation (X7), and Glazing Distribution (X8)** have weaker linear correlations but show varying average load values across their categories, suggesting some influence that might be better captured with more complex models or feature engineering.
- The simple linear regression models, using only X5, explained approximately 79.11% of the variance in 'Heating Load' and 80.24% of the variance in 'Cooling Load', indicating X5 is a significant predictor. However, there is still room for improvement by incorporating other features or using more advanced modeling techniques.