BUILDING ENERGY EFFICIENCY PREDICTION

Group 11

Sai Krishna Nandyala 1002144528
Vijay Ramesetty 1002170479

Contents:

Introduction:

With record-breaking temperatures across the world, it’s more important than ever to heat and cool our buildings efficiently. Whether you’re looking to lower your energy costs or reduce your carbon emissions, improving your building’s energy efficiency can help you save money and help the environment at the same time. Let’s take a look at a dataset that can help you with both.

Problem Statement:

The goal of this project is to develop a predictive model that can accurately estimate the energy efficiency of buildings based on various features. The dataset contains information about building features such as surface area, wall area, roof area, overall height, glazing area, and other relevant parameters. The target variable is the energy efficiency rating of the building.

Task Type:

Regression Modeling

Objective:

We'll download the data and clean it. Utilizing visualization tools can help us comprehend the data we are working with on a deeper level. After that, we'll develop the random forest and linear regression models. The Root Mean Squared Error will be used to assess the models (RMSE).

Finally, we will create a results table of the models and evaluate the results.

Why is this a Regression Problem?

The problem is considered a regression problem because the goal is to predict a continuous numerical outcome. In this case, the target variable is the energy efficiency rating of buildings, which is likely to be a continuous value. Regression analysis is used when the dependent variable is a continuous quantity, as opposed to a categorical or discrete one. If the variable is continuous, as in predicting a numerical measure like energy efficiency, it's a regression problem.

Data Source:

The dataset was created by Angeliki Xifara (Civil/Structural Engineer) and was processed by Athanasios Tsanas (Oxford Centre for Industrial and Applied Mathematics, University of Oxford, UK).The dataset has been downloaded from:
https://archive.ics.uci.edu/dataset/242/energy+efficiency

Dataset Description:

The buildings differ with respect to the glazing area, the glazing area distribution, and the orientation, amongst other parameters. The dataset comprises 768 samples and 8 features, aiming to predict two real valued responses. It can also be used as a multi-class classification problem if the response is rounded to the nearest integer. All the buildings have the same volume but different surface areas and dimensions. The materials used for each of the 8 elements are same for all building forms.

Features	Targets
Relative compactness – X1	Heating load – Y1
Surface area – X2	Cooling load – Y2
Wall area – X3	
Roof area – X4	
Overall height – X5	
Orientation – X6	
Glazing area – X7	
Glazing area Distribution – X8	

Features:

1. **Relative Compactness**: Compactness is defined as the ratio of the volume of the building to the surface area. It is measured as a dimensionless value between 0 and 1.
2. **Surface Area**: The total surface area of the building.
3. **Wall Area**: The total wall area of the building.
4. **Roof Area**: The total roof area of the building.
5. **Overall Height**: The height of the building.
6. **Orientation**: The orientation of the building.
7. **Glazing Area**: The total glazing area of the building.
8. **Glazing Area Distribution**: The distribution of glazing area among different faces of the building.

Targets:

The dataset contains two responses, **y1** and **y2**, which represent the **heating load** and **cooling load** of the building, respectively. The units for **y1** and **y2** are in **kWh/m²**.

Data Preprocessing:

Data preprocessing is a data mining technique in which raw data is transformed into an understandable form. Real-world data are often incomplete, inconsistent, and/or missing certain behavioral patterns or trends, and are likely to contain many errors. Data preprocessing is a way to solve such problems.

Missing Values:

Let's list out feature-wise count of missing values:

```
df.isnull().sum()
```

```
X1    0
X2    0
X3    0
X4    0
X5    0
X6    0
X7    0
X8    0
Y1    0
Y2    0
dtype: int64
```

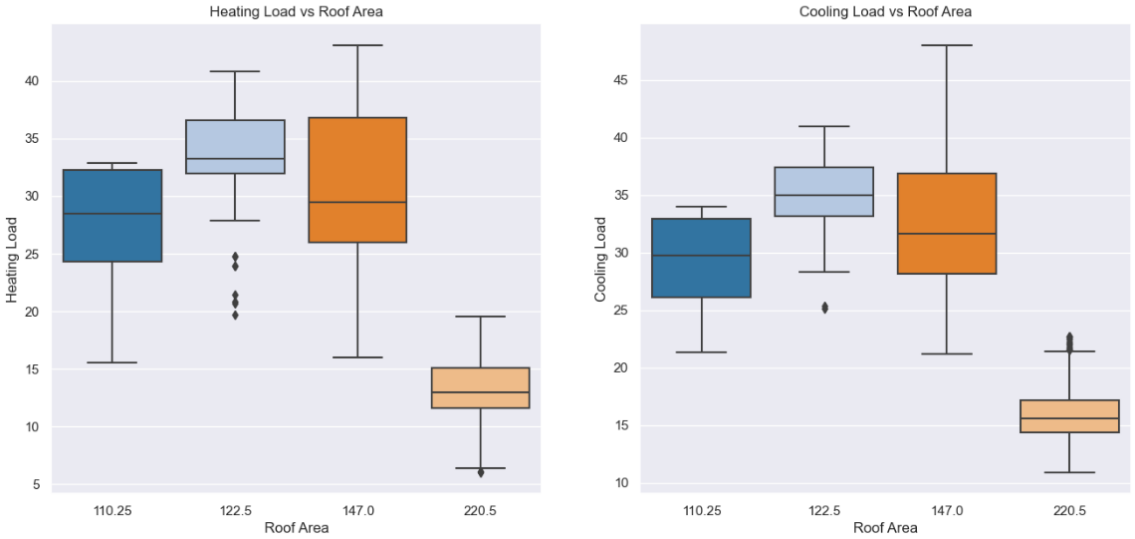
We don't have any missing values. The data is clean.

Outliers:

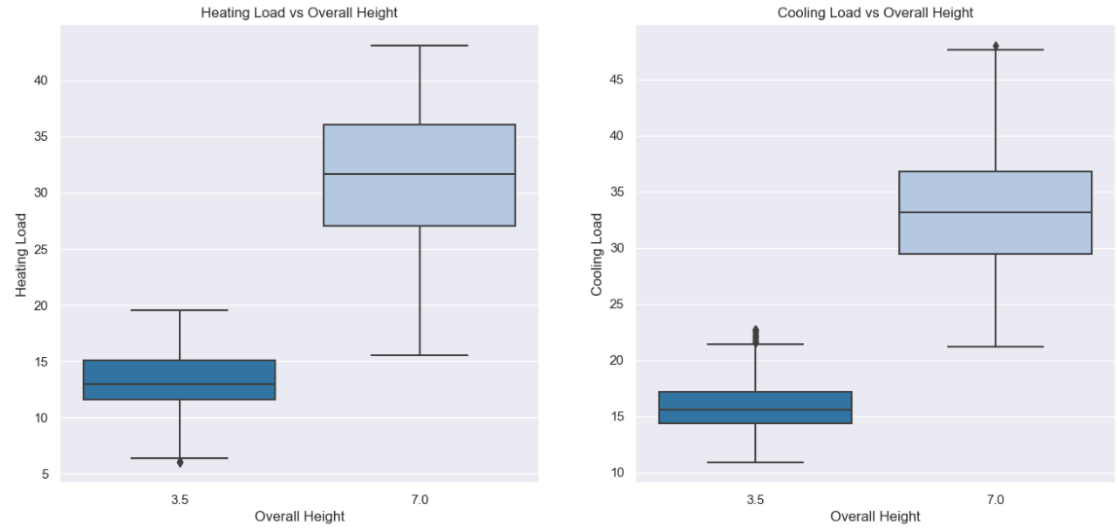
Outliers are data points that significantly differ from most of the data in a dataset. They can distort statistical analyses and machine learning models by introducing noise or bias. Identifying and handling outliers is essential for ensuring the robustness and accuracy of data-driven analyses. Techniques for outlier detection include statistical methods, visualization tools, and machine learning algorithms. Common approaches for handling outliers involve removing them, transforming the data, or using robust statistical measures. Addressing outliers is context-dependent, and the choice of method depends on the goals of the analysis and the characteristics of the dataset.

- The outliers can be seen for some categories in the plots drawn between targets and features.
- These outliers are seen for some categories in the features but not when the feature is considered entirely. So, we don't need to delete or handle the outliers.
- Below are the code and plots for the outliers seen between targets and features.

```
plt.figure(figsize=(16, 7))
plt.subplot(1,2,1)
plt.title('Heating Load vs Roof Area')
sns.boxplot(x = 'X4', y = 'Y1', data = df)
plt.xlabel('Roof Area')
plt.ylabel('Heating Load')
plt.subplot(1,2,2)
plt.title('Cooling Load vs Roof Area')
sns.boxplot(x = 'X4', y = 'Y2', data = df)
plt.xlabel('Roof Area')
plt.ylabel('Cooling Load')
plt.show()
```

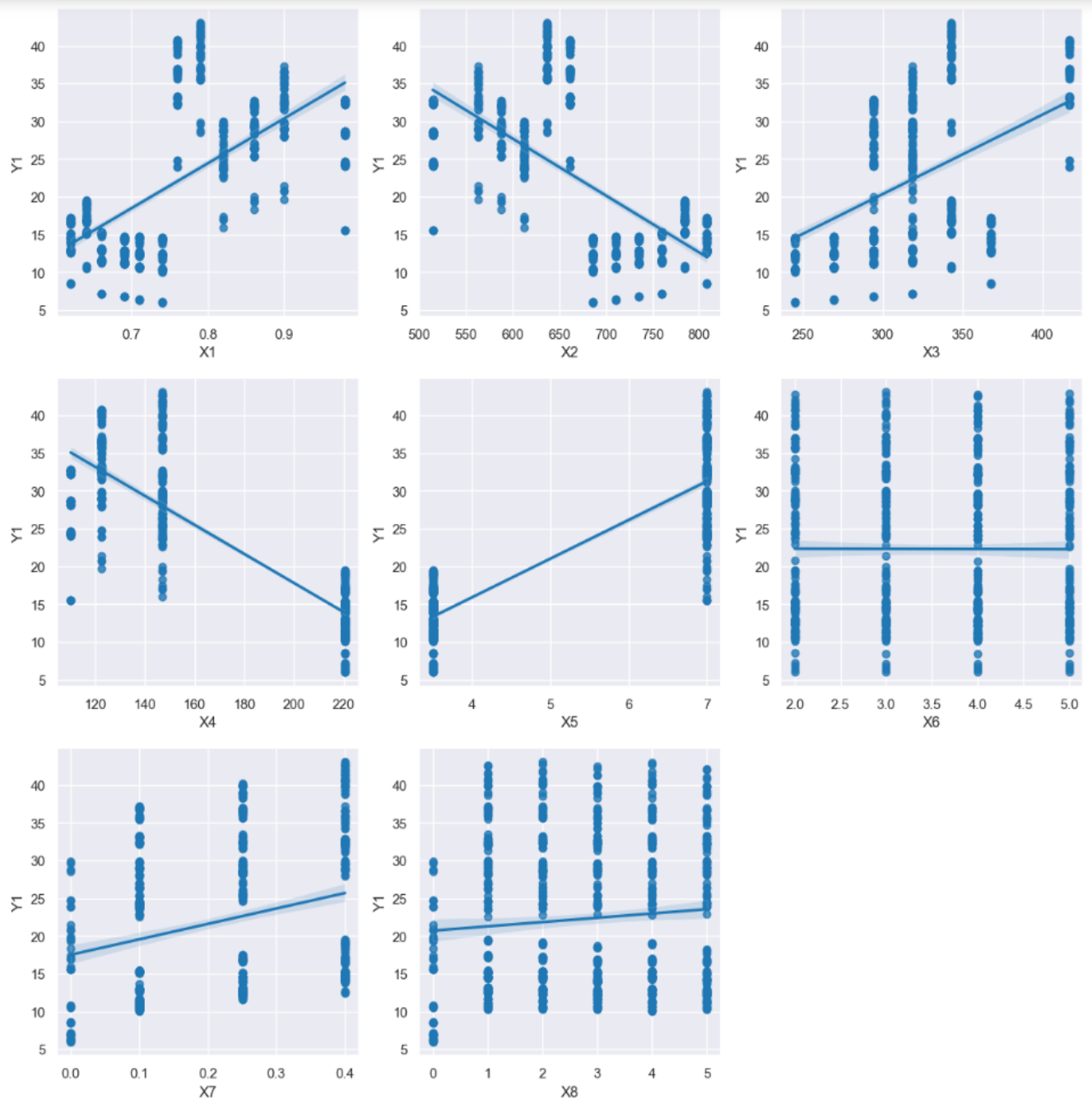



```
plt.figure(figsize=(16, 7))
plt.subplot(1,2,1)
plt.title('Heating Load vs Overall Height')
sns.boxplot(x = 'X5', y = 'Y1', data = df)
plt.xlabel('Overall Height')
plt.ylabel('Heating Load')
plt.subplot(1,2,2)
plt.title('Cooling Load vs Overall Height')
sns.boxplot(x = 'X5', y = 'Y2', data = df)
plt.xlabel('Overall Height')
plt.ylabel('Cooling Load')
plt.show()
```

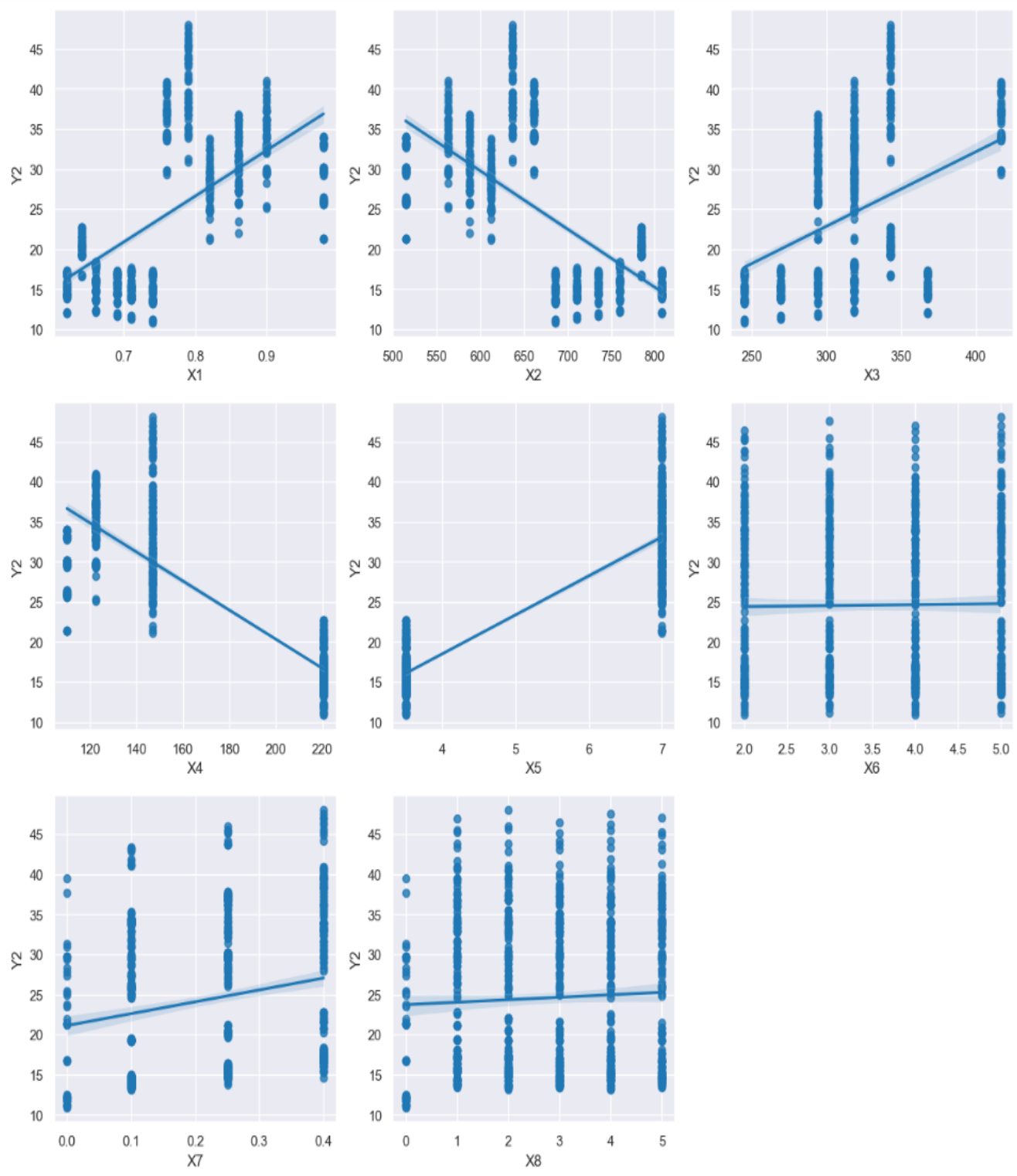


Data Visualization:

Visualizations complement statistical analysis by providing a graphical representation of data distributions, relationships, and statistical findings.We are presenting data in regression plots for visually representing the data correlation between target and each feature.



Y1 vs Features (X1 to X8)



Y2 vs Features (X1 to X8)

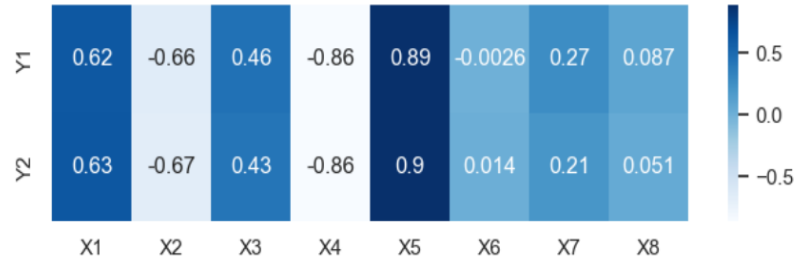
Exploratory Data Analysis (EDA):

To better understand the features and target variable, we will use Python to explore the data. Also, we will use a variety of visualization techniques to analyze the data and gather an outline of its key features.

Correlation Heat Map:

```
fig, ax = plt.subplots(figsize=(8,2))
sns.heatmap(df.corr().iloc[-2:,:8],annot=True,ax=ax,cmap="Blues")
```

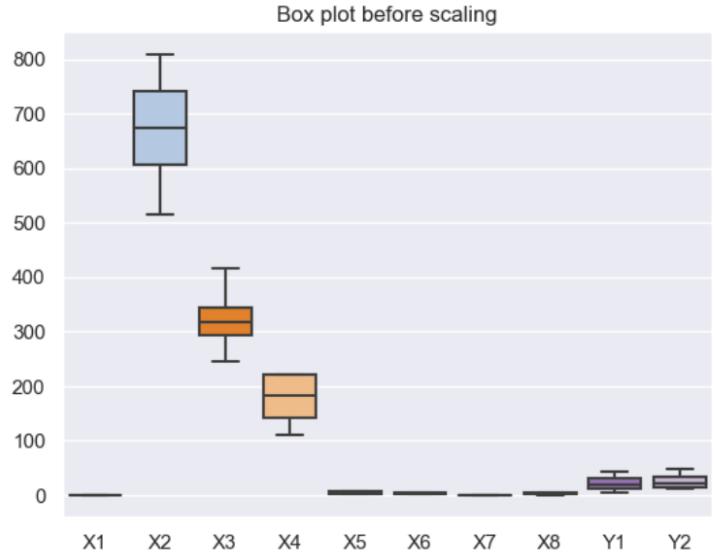
```
<Axes: >
```



- The correlation coefficient is a measure of the linear relationship between two variables. The closer the value is to 1, the stronger the positive relationship.
- The closer the value is to -1, the stronger the negative relationship. The closer the value is to 0, the weaker the relationship.
- Y1 and Y2 have the strongest correlation with X5(Overall Height) and weakest correlation with X2(Surface Area).

Data Scaling:

Data scaling is a preprocessing step that is often applied to the features of a dataset before training a machine learning model. The goal of scaling is to standardize or normalize the range of independent variables or features of the data. This can be important for certain algorithms that are sensitive to the scale of the input features.
Here is a boxplot of the data set before we scale. You can see the large differences in ranges between a few of the features.

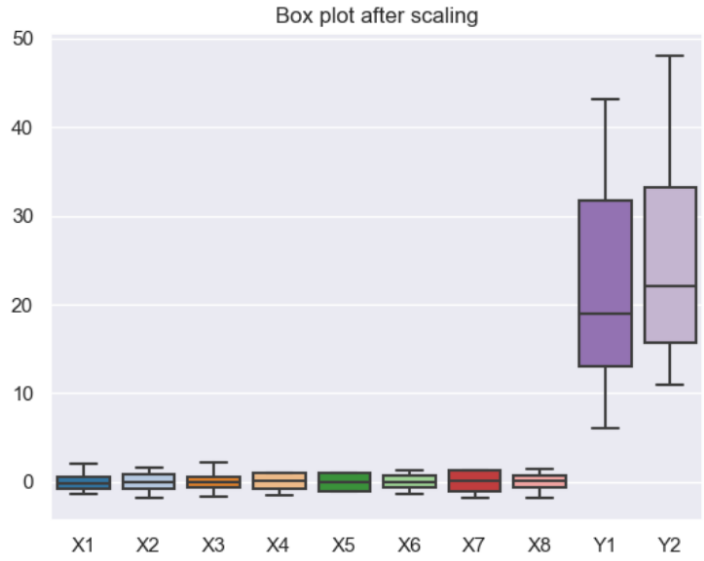


We are using standardization to scale the features.

Standardization (Z-score normalization):

- Formula: $X_{standardized} = (X - mean(X))/ std(X)$
- This method standardizes the data to have a mean of 0 and a standard deviation of 1. It is particularly useful when the features are normally distributed, and the algorithm relies on assumptions of normality.

Let's see a boxplot of the scaled dataset now. The features are scaled.



Model Building and Evaluation:

Train and Test Set:

The data has been split into 70% train set and 30% test set.

```
X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(X, y1, test_size = 0.3, random_state = 4528)
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(X, y2, test_size = 0.3, random_state = 4528)
```

Evaluation Metrics:

The model building process is not complete without evaluating the model's performance. Suppose we have a model prediction. So how can you tell if your predictions are correct? You can graph the results and compare them to the actual values. That is, you can calculate the distance between the predicted and actual values. The smaller the distance, the more accurate the prediction. Since this is a regression problem, you can evaluate your model using any of the following metrics:

1. **Mean Squared Error (MSE):**
    - MSE is the average of the squared differences between actual and predicted values. It penalizes larger errors more heavily.
2. **Mean Absolute Error (MAE):**
    - MAE is the average of the absolute differences between actual and predicted values. It is less sensitive to outliers compared to MSE.
3. **R-squared (R2) Score:**
    - R-squared measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges from 0 to 1, with higher values indicating better model fit.
4. **Root Mean Squared Error (RMSE):**
    - RMSE is the square root of the MSE, providing a measure in the same units as the target variable. It is another way to express the average prediction error.

Model Building:

1.Linear Regression:

- Linear regression is a statistical method that is used to model the relationship between a dependent variable (target) and one or more independent variables (features) by fitting a linear equation to the observed data.
- The case where there is only one independent variable is called simple linear regression, and when there are multiple independent variables, it is called multiple linear regression.

```
lr_heat = LinearRegression()
lr_heat.fit(X_train_1,y_train_1)
test_pred_heat = lr_heat.predict(X_test_1)
results_heat = pd.DataFrame(data=[["Linear Regression", *evaluate(y_test_1, test_pred_heat)]],
                            columns=['Heating Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])

lr_cool = LinearRegression()
lr_cool.fit(X_train_2,y_train_2)
test_pred_cool = lr_cool.predict(X_test_2)
results_cool = pd.DataFrame(data=[["Linear Regression", *evaluate(y_test_2, test_pred_cool)]],
                            columns=['Cooling Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(results_heat,"\n")
print(results_cool)
```

```
       Heating Model       MAE      MSE      RMSE  R2 Square
0  Linear Regression  1.979544  7.48544  2.735953   0.925638

       Cooling Model       MAE       MSE      RMSE  R2 Square
0  Linear Regression  2.143858  9.514455  3.084551   0.892247
```

2.Random Forest:

- Random Forest Regression is an ensemble learning method used for both classification and regression tasks. It belongs to the family of decision tree-based models and is an extension of the Random Forest algorithm for classification.
- In Random Forest Regression, multiple decision trees are trained on random subsets of the data, and their predictions are averaged to improve the overall predictive performance and reduce overfitting.

```
rf_heat = RandomForestRegressor(n_estimators = 100, random_state = 4528)
rf_heat.fit(X_train_1, y_train_1)
test_pred_heat = rf_heat.predict(X_test_1)
t1= pd.DataFrame(data=[["Random Forest Regressor", *evaluate(y_test_1, test_pred_heat) ]],
                            columns=['Heating Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])

rf_cool = RandomForestRegressor(n_estimators = 100, random_state = 4528)
rf_cool.fit(X_train_2, y_train_2)
test_pred_cool = rf_cool.predict(X_test_2)
t2= pd.DataFrame(data=[["Random Forest Regressor", *evaluate(y_test_2, test_pred_cool) ]],
                            columns=['Cooling Model', 'MAE', 'MSE', 'RMSE', 'R2 Square'])
print(t1,"\n")
print(t2)
```

```
             Heating Model       MAE     MSE      RMSE  R2 Square
0  Random Forest Regressor  0.360672  0.2826  0.531602   0.997193

             Cooling Model       MAE       MSE      RMSE  R2 Square
0  Random Forest Regressor  0.960788  2.591452  1.609799   0.970651
```

Model Comparison:

```
results_heat
```

	Heating Model	MAE	MSE	RMSE	R2 Square
0	Linear Regression	1.979544	7.48544	2.735953	0.925638
1	Random Forest Regressor	0.360672	0.28260	0.531602	0.997193

```
results_cool
```

	Cooling Model	MAE	MSE	RMSE	R2 Square
0	Linear Regression	2.143858	9.514455	3.084551	0.892247
1	Random Forest Regressor	0.960788	2.591452	1.609799	0.970651

- The model with the best performance was the random forest model. It has the highest R-squared score and low MAE and MSE which indicates better model performance.
- The model with the highest RMSE was the linear regression one. It has the low R-squared score and high MAE and MSE which indicates poor model performance.

Conclusion:

- Using a dataset of building characteristics and heating and cooling loads, we were able to forecast the heating and cooling demand of buildings.
- To make predictions, we employed supervised machine learning techniques. There were two regression models in total: random forest and linear regression.
- With an RMSE of 0.53, the random forest model proved to be the most accurate model for estimating the heating load.
- With an RMSE of 1.61, the random forest model proved to be the most accurate model in predicting the cooling load.

References:

- Numpy Documentation: https://numpy.org/devdocs/user/quickstart.html
- User guide for Pandas: https://pandas.pydata.org/docs/user_guide/index.html
- Seaborn gallery: https://seaborn.pydata.org/examples/index.html
- Matplotlib gallery: https://matplotlib.org/3.1.1/gallery/index.html
- Energy Efficiency Dataset https://www.kaggle.com/elikplim/eergy-efficiency-dataset/notebooks