Question 2

At first, the data is plotted to check the data distribution.

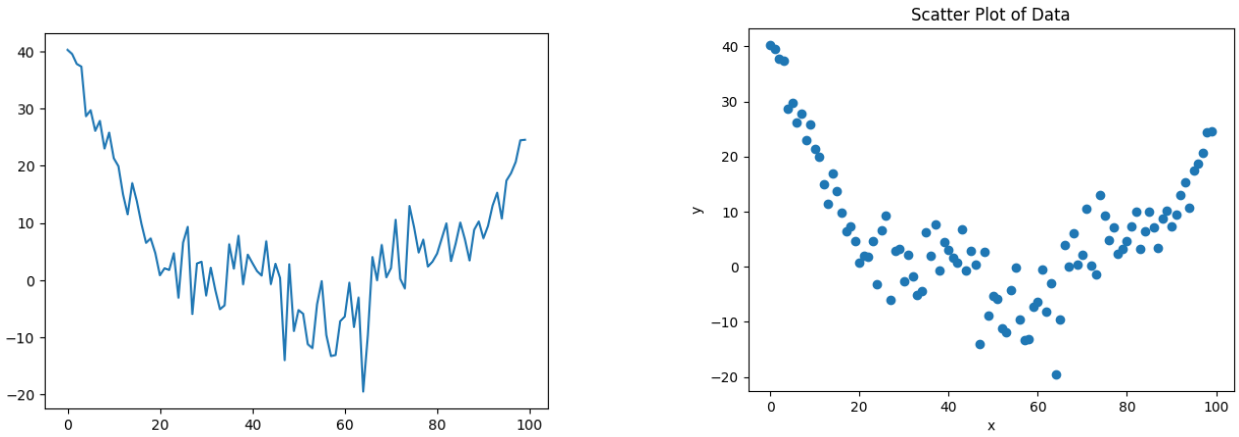

Figure 1: Left: Line Plot of Data. Right: Scatter Plot of Data.

Part 1

I allocated 25% of the dataset to the test set and the rest to the train set. Additionally, the random state value is set to 73. In the figure below, the data is displayed, indicating which data points are assigned to each set.

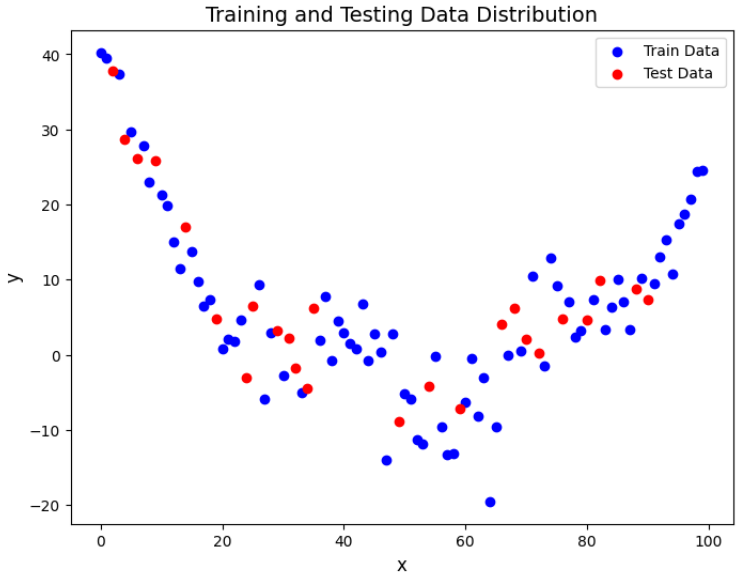

Figure 2: Training and Testing Data Distribution.

Part 2-Metrics for Regression Model Evaluation

We compare the actual values and predicted values to calculate the accuracy of a regression model. Evaluation metrics provide a key role in the development of a model, as it provides insight to areas that require improvement.

1. Mean Squared Error (MSE)

The Mean Squared Error (MSE) is a commonly used metric that measures the average squared difference between the predicted values ($\hat{y}_i$) and the actual values (y_i). It is particularly sensitive to large errors because the differences are squared. A smaller MSE indicates that the model's predictions are closer to the actual values, while a larger MSE reflects greater prediction errors. This metric is widely used in regression problems, where penalizing larger errors more heavily is desirable.

The MSE is calculated using the formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

2. Root Mean Squared Error (RMSE)

The Root Mean Squared Error (RMSE) is the square root of the Mean Squared Error. By taking the square root, RMSE transforms the error into the same unit as the target variable, making it more interpretable than MSE. RMSE is sensitive to large errors due to the squaring in its calculation, similar to MSE. However, it provides a clearer understanding of the average prediction error in the context of the problem's domain.

The RMSE is calculated using the formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

3. Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) measures the average of the absolute differences between the predicted values and the actual values. Unlike MSE, MAE treats all errors equally, regardless of their size. This characteristic makes MAE less sensitive to outliers compared to MSE or RMSE. MAE is a straightforward metric that directly reflects the magnitude of prediction errors.

The MAE is calculated using the formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

4. R^2 (Coefficient of Determination)

The R^2 metric, also known as the Coefficient of Determination, quantifies the proportion of variance in the target variable that is explained by the model's predictions. It ranges from 0

to 1, where 1 indicates a perfect fit and 0 suggests that the model performs no better than simply predicting the mean of the target variable. A higher R^2 value signifies a better-fitting model. However, R^2 does not directly measure prediction accuracy, and a high R^2 does not always guarantee a good model.

The R^2 is calculated using the formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the actual values.

Part 3, First-Degree Linear Degression Model

In this section, using the dataset I had, I trained a first-order linear regression model without using ready-made functions. The results and graphs are presented below.

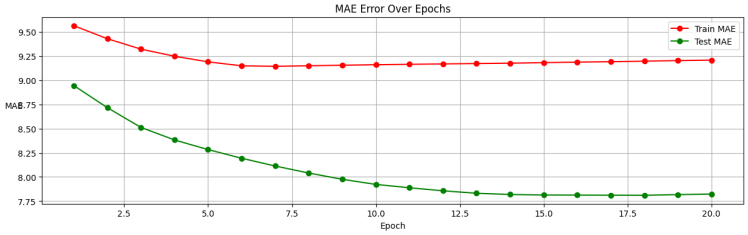

Figure 3: Mean Absolute Error (MAE) Trend During Training Over Epochs

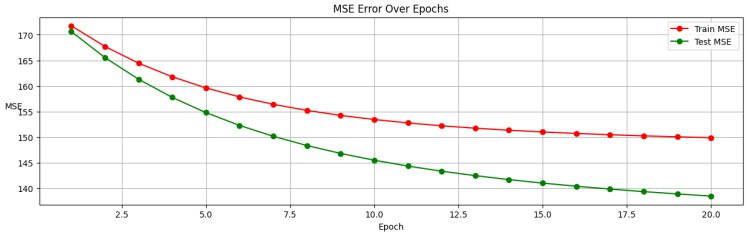

Figure 4: Mean Squared Error (MSE) Trend During Training Over Epochs

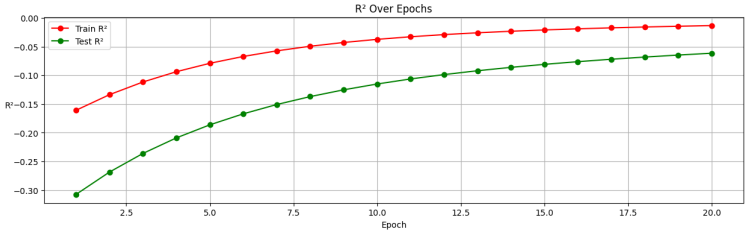

Figure 5: R^2 Score Trend During Training Over Epochs

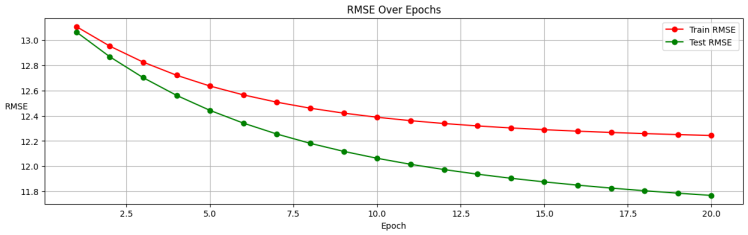

Figure 6: Root Mean Squared Error (RMSE)Trend During Training Over Epochs

```
Final Slope (w): 0.6523521081462805
Final Intercept (b): 4.823074903191027

Final Errors:
Train MSE: 149.9029811616927
Test MSE: 138.51410104497046
Train RMSE: 149.9029811616927
Test RMSE: 138.51410104497046
Train MAE: 9.206567509482719
Test MAE: 7.82321532026446

Final R² Scores:
Train R²: -0.013226227874048035
Test R²: -0.06142059017691137
```

Figure 7: Evaluation results

From the graphs and evaluation results, we can see that the model performance is very poor. Although the error decreases during the training process, it still does not perform well. Even the value of R^2 has become negative, which indicates a very poor performance of this model. The results of the 4 evaluations show that the first-order model cannot perform well on this dataset.

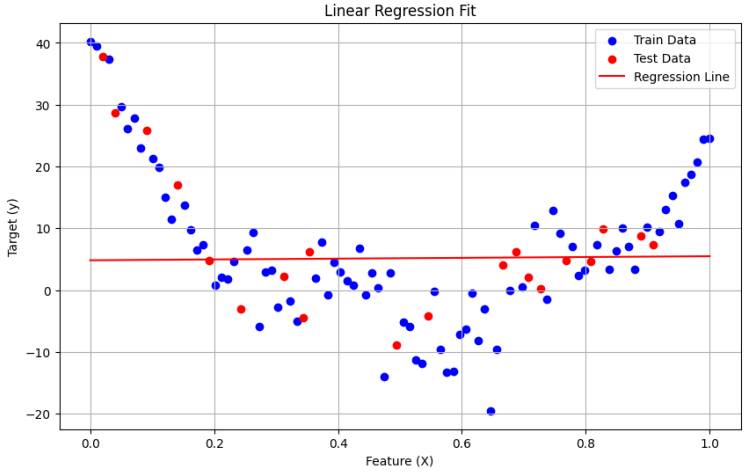

Figure 8: The obtained line and the training and testing data

As can be seen in the figure, a linear first-order model is not appropriate for this data set. The data set follows a nonlinear pattern and a more complex model should be used.

Part 4

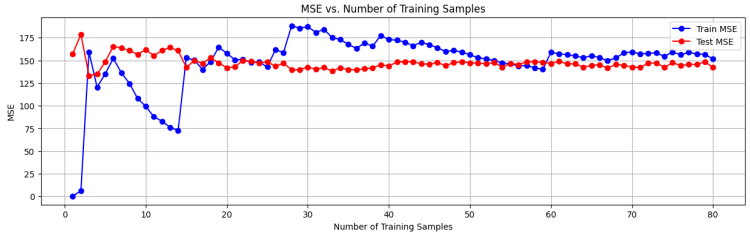

Figure 9: Mean Squared Error (MSE)

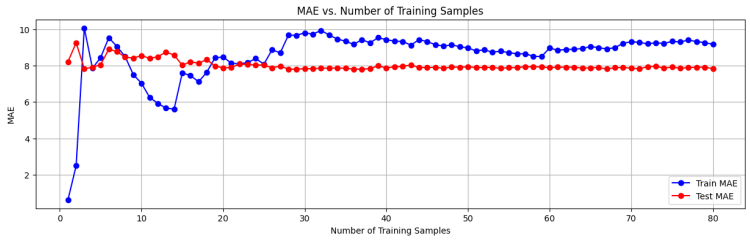

Figure 10: Mean Absolute Error (MAE)

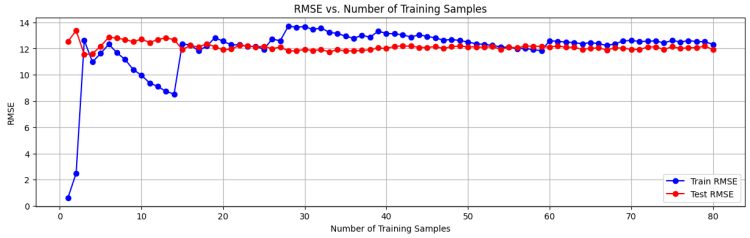

Figure 11: Root Mean Squared Error (RMSE)

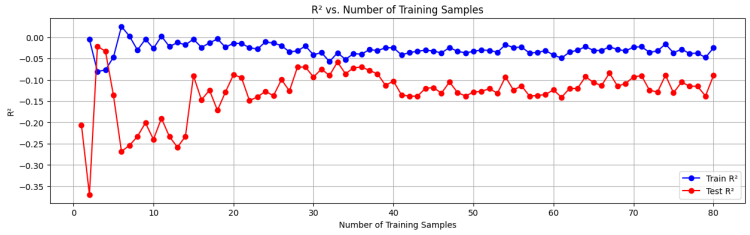

Figure 12: R^2 Score Trend

```
Final Slope (w): 2.037352418865679
Final Intercept (b): 3.230129336708424


Final Errors:
Train MSE: 154.5487
Test MSE: 148.1319

Train RMSE: 12.4318
Test RMSE: 12.1709

Train MAE: 9.1760
Test MAE: 7.9072

Final R² Scores:
Train R²: -0.0446
Test R²: -0.1351
```

Figure 13: Evaluation Results

As the number of data increases, the accuracy of the model on the training data decreases but on the test data it almost increases, but after a certain point, the accuracy of the model does not change. The final results still show that the model performance is poor.

Part 5

To answer the question of whether we can bring the capabilities of an Artificial Intelligence model to those of a human, need to consider two key concepts in Atificial Intelligence: model capacity and data quality.

1. Model Capacity

Model capacity refers to the complexity a model can learn. If our model is sufficiently complex, it is theoretically capable of learning more intricate patterns and achieving better accuracy. However, if the model's capacity is limited, adding more data alone may not reduce the error to the desired level.

2. Quality and Quantity of Data newline Increasing the amount of data can help reduce variance error. However, if the model's initial training error is high (10 in this case), it is likely suffering from high bias. High bias means the model does not have enough capacity or flexibility to learn the patterns effectively.

Considering these two points, if the data quality is good and the model has a high capacity for learning, we can bring the accuracy of the AI model to human accuracy. But these two points must be observed at the same time. For example, here the number of data increases but because the capacity of the model is low, we cannot achieve good accuracy.

Part 6

In this section, we add one term to the model at a time to see if the accuracy increases as the model capacity increases.

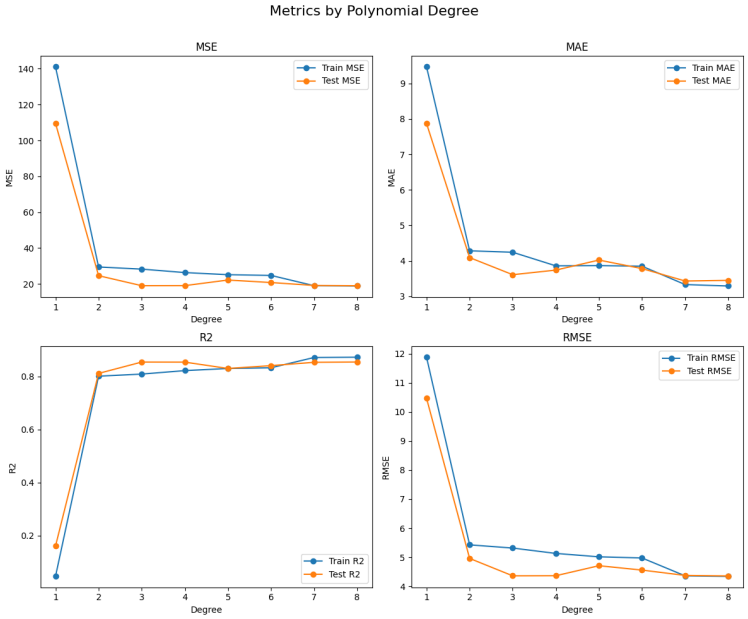

Figure 14: Metrics by Polynomial Degree

As you can see, increasing the number of terms almost increases the accuracy of the model, of course, we have the best performance in term 3 and then the performance drops slightly, but after that the accuracy of the model increases and when we reach term 8, the accuracy of the model reaches its maximum. It should be noted that increasing the capacity of the model does not necessarily increase the accuracy and can lead to overfitting. It should be noted that complexity does not necessarily lead us to the best results and sometimes the answer lies in simplicity.

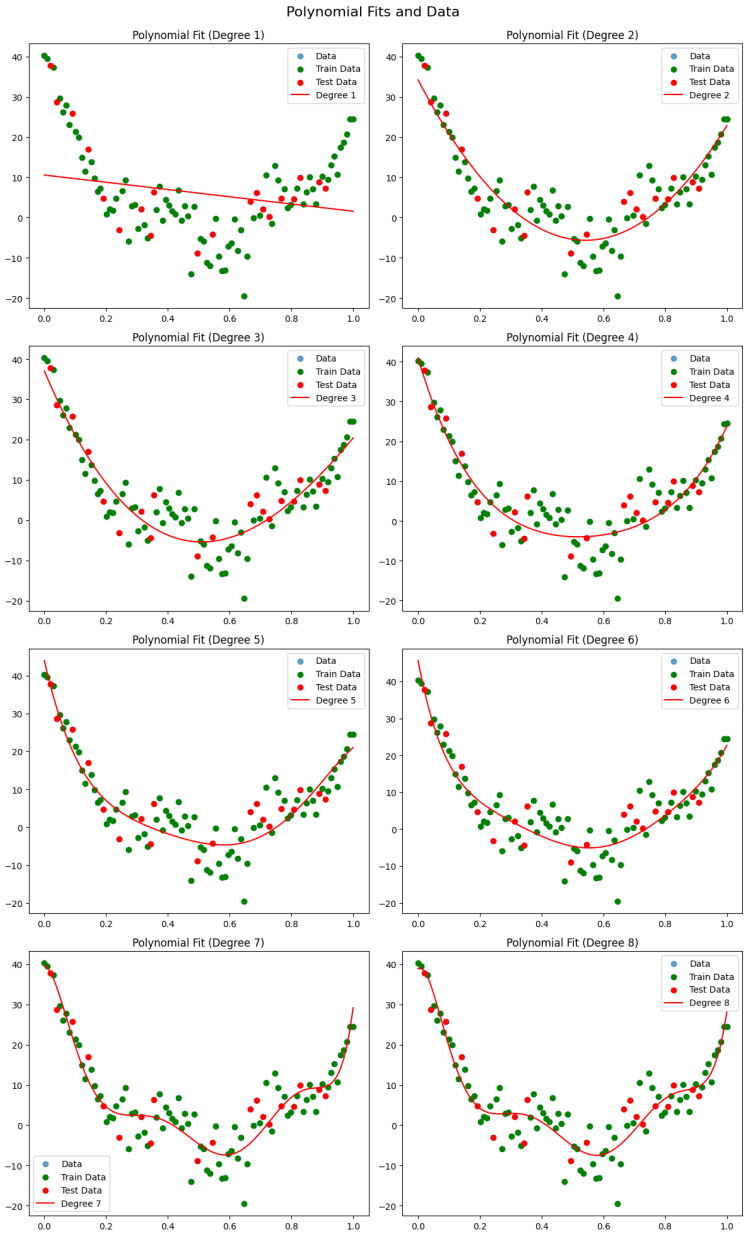

Figure 15: Polynomial Fits and Data

In the figure above, we have shown the lines obtained from the models in R^2 space. According to the graphs, we can say that by increasing the terms, the sensitivity of the model can also increase.

Degree	Train MSE	Test MSE	Train MAE	Test MAE
1	141.090723	109.564462	9.468924	7.877281
2	29.447861	24.603269	4.281312	4.089658
3	28.265974	19.024115	4.239992	3.607097
4	26.336162	19.056088	3.856924	3.736937
5	25.154638	22.178963	3.863441	4.019999
6	24.757821	20.784772	3.846692	3.779226
7	19.015115	19.145629	3.329026	3.429129
8	18.855123	18.994026	3.288824	3.448400

Table 1: Metrics Summary for Polynomial Models (Part 1: MSE and MAE)

Degree	Train R2	Test R2	Train RMSE	Test RMSE
1	0.046338	0.160418	11.878162	10.467304
2	0.800956	0.811467	5.426588	4.960168
3	0.808944	0.854220	5.316575	4.366164
4	0.821988	0.853975	5.131877	4.365328
5	0.829947	0.830045	5.015440	4.709455
6	0.832657	0.840728	4.975723	4.559032
7	0.871473	0.853289	4.360632	4.375572
8	0.872554	0.854451	4.342249	4.358214

Table 2: Metrics Summary for Polynomial Models (Part 2: R2 and RMSE)

As you can see, the 3rd degree and 8th degree models have almost the same performance. Especially when testing the model on the test data, the accuracies are very close. We can conclude that if we were facing resource shortage problem when presenting the final model, we can use the 3rd degree model instead of the 8th degree model to reduce the computational complexity.

Part 7

In this section, I have used Grid Search to find the best parameters.

Grid Search is a systematic approach totuning hyperparameters in machine learning models. It involves defining a grid of possible values for the model's hyperparameters and exhaustively searching through all combinations to identify the best configuration. The evaluation of each combination is usually done using cross-validation to ensure robust selection. This method is computationally expensive but guarantees exploration of all specified hyperparameter values, making it effective for models with smaller hyperparameter spaces. Its key advantages are its simplicity and accuracy, while its main disadvantage is its high computational cost for large networks.

DecisionTreeRegressor

The DecisionTreeRegressor is a machine learning model that predicts continuous values by recursively splitting the data into smaller regions. At each step, it selects the best feature and threshold to divide the data, aiming to minimize the error in each resulting subset. This process continues until a stopping criterion, such as maximum tree depth or minimum samples per leaf, is met.

Each leaf node of the tree represents a region of the feature space, and the model predicts the average target value of the samples within that region. Decision trees are capable of modeling non-linear relationships and are easy to interpret, as they can be visualized and analyzed step by step.

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
Best Parameters:
  max_depth: 5
  max_features: None
  min_samples_leaf: 1
  min_samples_split: 2

Metrics:
  Train MSE: 13.6216
  Test MSE: 37.7372
  Train MAE: 2.5402
  Test MAE: 4.8796
  Train RMSE: 3.6907
  Test RMSE: 6.1431
  Train R2: 0.9079
  Test R2: 0.7108
```

Figure 16: Evaluation Results of Decision Tree Regressor

Parameters

- **max_depth: 5**
  This parameter specifies the maximum depth of the decision tree. A depth of 5 means the tree can split the data up to 5 levels, limiting its complexity and helping prevent overfitting.

- **max_features: None**
  This indicates that all features are considered when looking for the best split at each node. If it were set to `sqrt` or `log2`, only a subset of features would be used, which could reduce overfitting but might decrease model performance.

- **min_samples_leaf: 1**
  This specifies the minimum number of samples required to be in a leaf node. A value of `1` means the leaf nodes can contain a single data point. Higher values reduce the complexity of the tree and help prevent overfitting.

- **min_samples_split: 2**
  This defines the minimum number of samples required to split an internal node. A value of `2` allows splitting if at least 2 samples are present in the node. Increasing this value would make the tree less sensitive to noise and reduce overfitting.

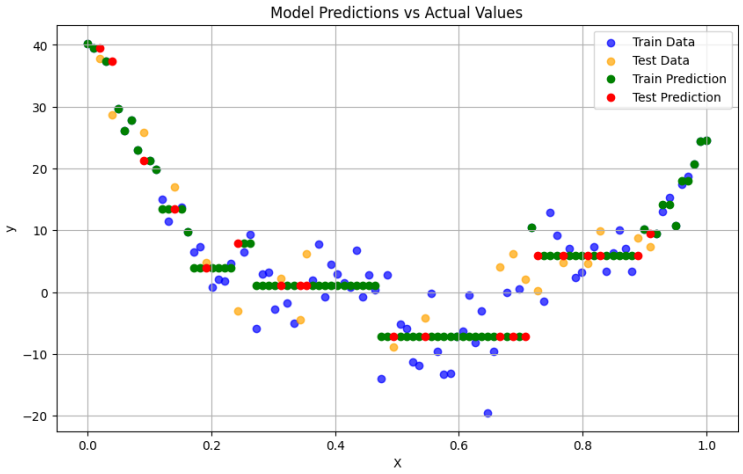

Figure 17: Comparing model output and actual values

From the evaluation results and the graph output, we understand that the model is slightly overfitted.

Polynomial PolynomialFeatures

PolynomialFeatures is a way to generate new features by raising existing features to a specified polynomial degree. This transformation allows machine learning models, such as linear regression, to model non-linear relationships between the input features and the target variable.

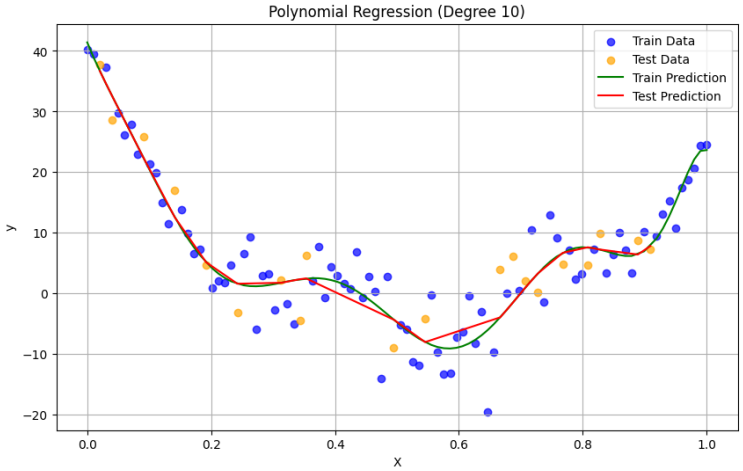

Figure 18: Comparing model output and actual values

```
{'Train MSE': 16.411226453247608,
 'Test MSE': 16.22508017580954,
 'Train MAE': 3.014308935101017,
 'Test MAE': 3.380208599519581,
 'Train R2': 0.8890730194613355,
 'Test R2': 0.8756687294221096}
```

Figure 19: Comparing model output and actual values

From the evaluation results and the graph, we can see that the model performance is very good. The accuracy of the model is still close to the test and training data and we can say that the model is not overfitted. It is also clear from the graph that the line is well fitted.

```
] terms = [f"{coeff:.4f}*x^{i}" for i, coeff in enumerate(coefficients) if abs(coeff) > 1e-3]
  polynomial_equation = " + ".join(terms)
  polynomial_equation = f"y = {intercept:.4f} + " + polynomial_equation[7:]

  print("Simplified Polynomial Equation:")
  print(polynomial_equation)

  Simplified Polynomial Equation:
  y = 41.4165 + 55*x^1 + 1442.1383*x^2 + -17202.8541*x^3 + 91869.3870*x^4 + -163501.6392*x^5 + -165686.9259*x^6 + 1055026.5375*x^7 + -1593621.1166*x^8 + 1062281.0887*x^9 + -270368.4414*x^10
```

Figure 20: Obtained polynomial

XGBOOST Regression

XGBoost (Extreme Gradient Boosting) is an efficient and scalable implementation of gradient boosting. It's widely used for supervised learning problems, including regression, due to its ability to model complex relationships while being computationally efficient. XGBoost builds an ensemble of decision trees where each tree corrects the errors of the previous ones, optimizing the objective function.

Paremeters:

```
xgb_model = XGBRegressor(

    n_estimators=20,  # Number of trees
    learning_rate=0.1,  # Step size shrinkage
    max_depth=4,  # Maximum depth of a tree
    reg_alpha=0.1,  # L1 Regularization
    reg_lambda=0.1,  # L2 Regularization
    random_state=73  # For reproducibility
)
```

Figure 21: Parameters of the model

- **n_estimators=20**
  Specifies the number of boosting rounds (trees) the algorithm will train. A higher number of estimators generally improves performance, but it can lead to overfitting if too high.

- **learning_rate=0.1**
  The learning rate (also called *eta*) shrinks the contribution of each tree by this value. A lower learning rate slows down the training process but helps the model generalize better. It requires more trees (`n_estimators`) to achieve good performance.

- **max_depth=4**
  The maximum depth of each tree in the model. A higher value increases the complexity of the model, allowing it to capture more details in the data, but it also increases the risk of overfitting.

- **reg_alpha=0.1**
  The L1 regularization term on weights. It adds a penalty for the absolute values of weights in the objective function. It helps reduce overfitting by forcing the model to prefer smaller, simpler weights, effectively pruning irrelevant features.

- **reg_lambda=0.1**
  The L2 regularization term on weights. It penalizes the squared values of weights in the objective function.

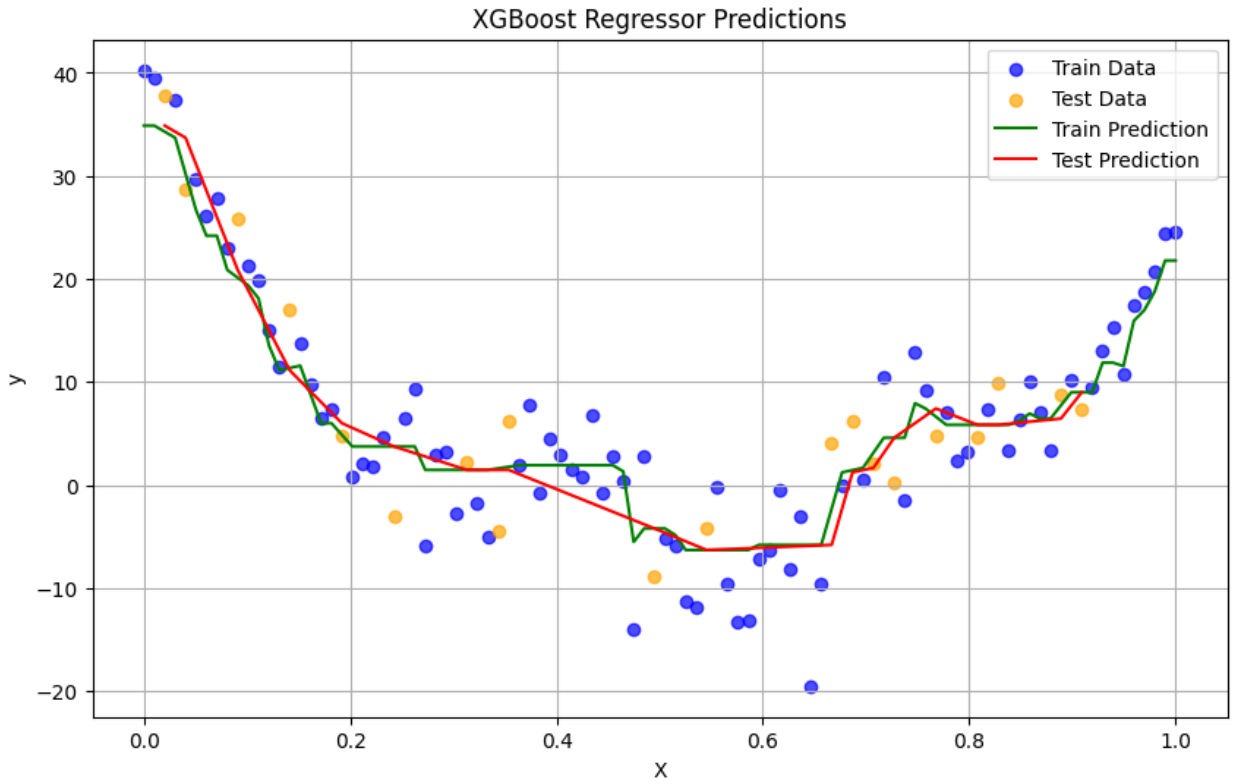

Figure 22: Obtained polynomial

```
{'Train MSE': 14.241995636410383,
 'Test MSE': 19.92941180971756,
 'Train MAE': 2.9598514300070455,
 'Test MAE': 3.8237131046310595,
 'Train R2': 0.9037353133056544,
 'Test R2': 0.8472827828692953,
```

Figure 23: Model evaluation

The evaluation results show that the model has good accuracy and does not suffer from overfitting. However, despite the high capacity of this model, the results obtained from polynomial regression were the best result.

Part 8 - Reguralization

Regularization is a technique used in machine learning to prevent overfitting by adding a penalty to the model's complexity. It discourages the model from fitting too closely to the training data, which could lead to poor generalization to new data.

In the context of linear regression, regularization modifies the loss function by adding a penalty term to control the magnitude of the model's coefficients. The goal is to balance the trade-off between the fit of the model to the training data and the complexity of the model.

There are two commonly used types of regularization:

1. Lasso Regularization (L1)

Lasso regularization adds the absolute values of the coefficients to the loss function. It encourages sparsity, meaning it tends to set some coefficients to zero, effectively performing feature selection.

The formula for Lasso Regression is:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 + \lambda \sum_{j=1}^{n} |\theta_j|$$

λ is the regularization parameter that controls the strength of regularization. A higher value means more regularization. α in Elastic Net determines the mix of L1 and L2 regularization (e.g., $\alpha = 0.5$ gives equal weight to both).

2. Ridge Regularization (L2)

Ridge regularization adds the squared values of the coefficients to the loss function. It encourages small coefficients but does not set them exactly to zero.

The formula for Ridge Regression is:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 + \lambda \sum_{j=1}^{n} \theta_j^2$$

3. Elastic Net Regularization

Elastic Net regularization combines L1 and L2 regularization. It is useful when there are multiple features that are correlated.

The formula for Elastic Net is:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 + \lambda \left(\alpha \sum_{j=1}^{n} |\theta_j| + (1 - \alpha) \sum_{j=1}^{n} \theta_j^2 \right)$$

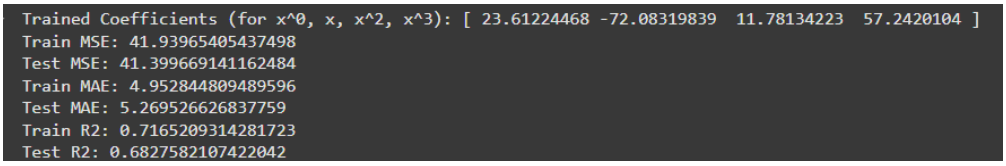

Figure 24: Model evaluation : Model with L1 Regularization

The training and testing R2 values indicate that the model explains 71.6 of the variance in the training data and 68.2 in the testing data. This is a reasonable fit, showing the model generalizes relatively well.

Regularization Effect:

Model Fit

The training and testing R^2 values indicate that the model explains approximately 71.6% of the variance in the training data and 68.2% in the testing data. This is a reasonable fit, showing that the model generalizes relatively well.

Regularization Effect

$L1$ regularization has helped by shrinking the coefficients and potentially reducing overfitting.

Errors

The Mean Squared Error (MSE) and Mean Absolute Error (MAE) are comparable for training and testing datasets, further indicating that the model generalizes well.