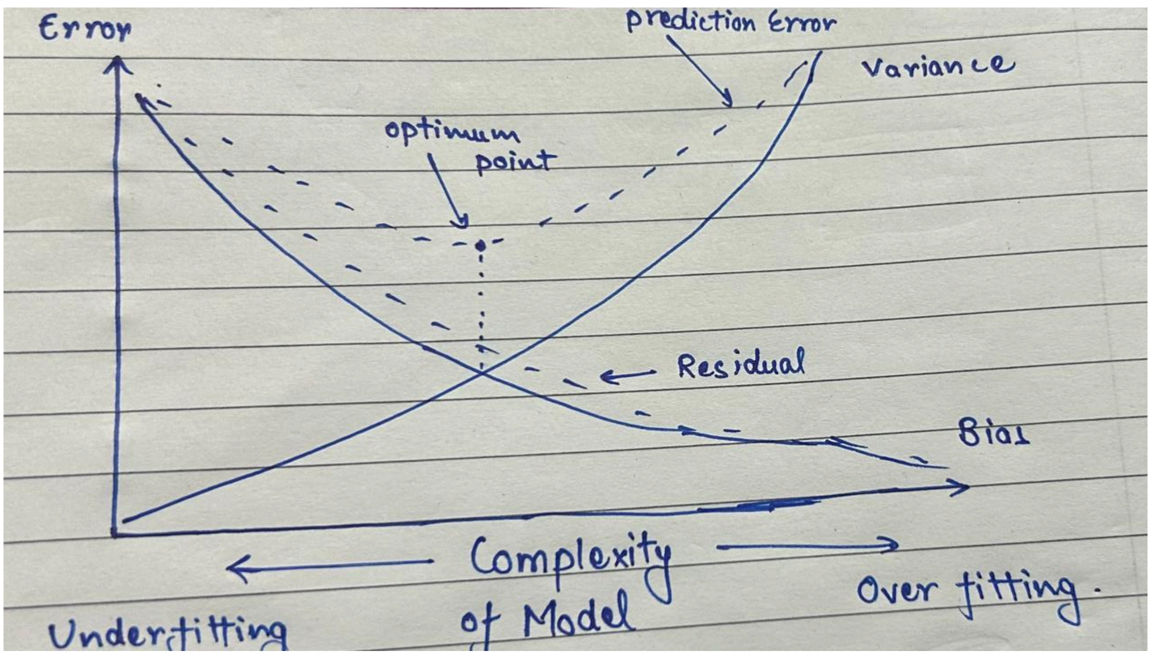
Report

Section A

a) As we increase the complexity of a machine-learning model, there's a higher risk of **overfitting**. This means the model might learn the training data too well, leading to poor performance on new, unseen data. A complex model with many features or polynomial terms can easily memorize the training set, resulting in a high training accuracy but low generalization. Techniques like regularization, cross-validation, and early stopping can help prevent overfitting and improve the model's performance on new data.
High Variance: These models are highly sensitive to the specific training data they see. Different training sets can lead to significantly different model behaviors, resulting in high variance.
Low Bias: Complex models, when not overfitting, can generally capture complex patterns in the data. This means they have low bias, as they are not making overly simplistic assumptions.



b) Classifying a mail is assumed to be positive
True positive= 200
False positive: 20
True Negative= 730
False Negative= 50

Total correct predictions = = 200 + 730 = 930
Precision = TP / (TP + FP) = 200 / (200 + 20) = 0.9091

Recall = TP / (TP + FN) = 200 / (200 + 50) = 0.80
Accuracy =(TP + TN)/ (TP + FP + TN + FN) =(200 + 730) /(200 + 20 + 730 + 50) = 0.93
Negative predicted value = TN / (TN + FN) = 730 / (730 + 50) ≈ 0.9355
Specificity = TN / (TN + FP) = 730 / (730 + 20) = 0.9730
F1-Score = 2 * (Precision * Recall) / (Precision + Recall) = 2 * (0.9091 * 0.80) / (0.9091 + 0.80) ≈ 0.85
The model performs impressively with high precision, recall, accuracy, and specificity. The strong F1-Score shows a good balance between precision and recall, indicating effective classification with minimal errors. If we aim to improve specific aspects like recall, we might adjust model parameters or try different algorithms. Overall, the model is highly effective.

c) $\Sigma x = 52$
$\Sigma y = 285$
$\Sigma x^2 = 694$
$\Sigma xy = 3850$
a=$(\overline{xy} - \overline{x}.\overline{y})/(\overline{x^2} -.\overline{x}^2)$
b= $\overline{y} - a.\overline{x}$

By using these formulas:
a=5.78 and b= −3.11
equation for linear regression:
y= ax+b => **y= 5.78x- 3.11**
Now to predict y putting x=12,
y= 66.25

d) Let us take an example of number of room vs price prediction example,

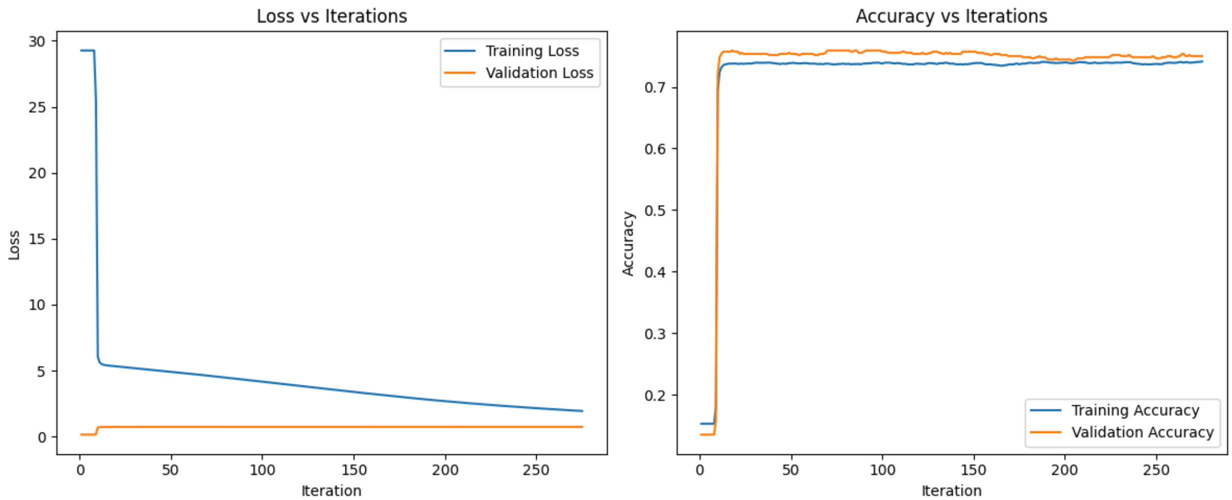
Number of rooms	Price
1	150k
2	300k
3	250k
4	325k
5	400k

F1: High-Capacity Models (like polynomial regression) often have lower empirical risk on training data due to their flexibility, but they might not generalize well to unseen data.
F2: Low-Capacity Models (like linear regression or regularized models) may have higher empirical risk on training data but generally perform better on new data due to

their simplicity and reduced risk of overfitting.

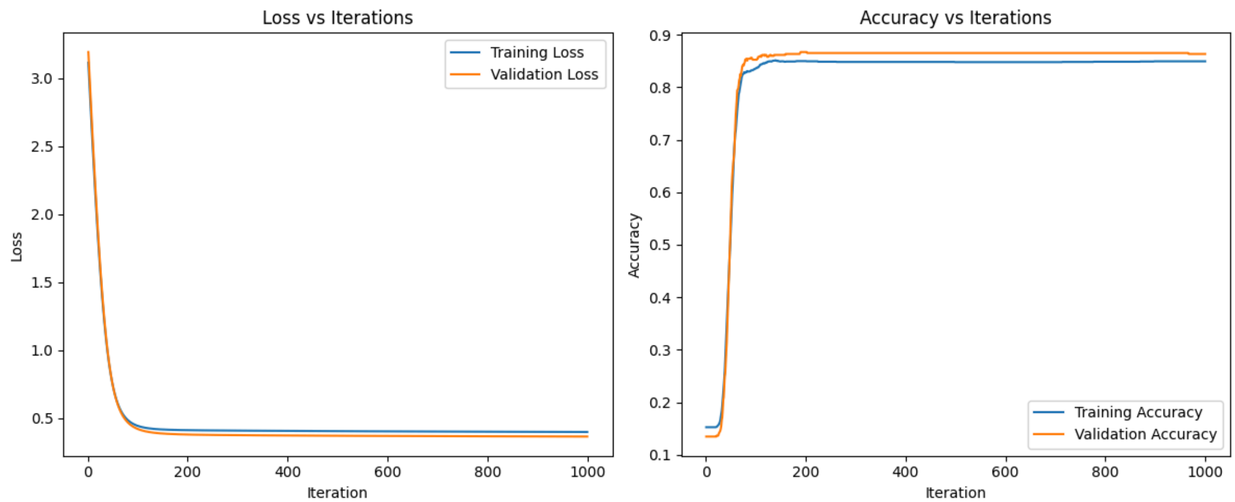
Section B

(a) (3 marks) Implement Logistic Regression using Batch Gradient Descent. Plot training loss vs. iteration, validation loss vs. iteration, training accuracy vs. iteration, and validation accuracy vs. iteration. Comment on the convergence of the model. Compare and analyze the plots



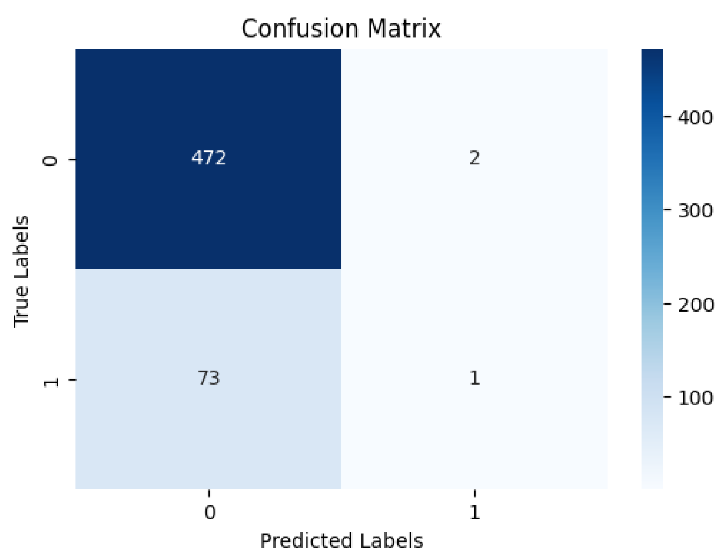
Training Loss: The blue curve shows that the training loss decreases rapidly in the first few iterations and then continues to decrease slowly, indicating that the model is learning effectively.
Validation Loss: The orange curve remains very close to zero throughout the training process. This could imply that the validation loss is significantly lower than the training loss, which might be due to overfitting.
Training Accuracy: The blue curve shows that the training accuracy starts low but rapidly increases, stabilizing around an accuracy of 0.72 after the first few iterations.
Validation Accuracy: The orange curve shows a similar trend, but it appears slightly higher than the training accuracy, stabilizing near 0.75.
Training loss is decreasing and **accuracy is improving**, meaning the model is learning effectively.

(b) (2 marks) Investigate and compare the performance of the model with different feature scaling methods: Min-max scaling and No scaling. Plot the loss vs. iteration for each method and discuss the impact of feature scaling on model convergence.



These plots are obtained after applying Min-Max scaling to the data. With scaling, the model is able to learn effectively at a significantly higher learning rate compared to when no scaling is applied. Despite the faster learning process, the overall convergence behavior remains similar to what was observed without scaling i.e. loss and accuracy and converge at almost the same value with or without scaling.

(c) (2 marks) Calculate and present the confusion matrix for the validation set. Report precision, recall, F1 score, and ROC-AUC score for the model based on the validation set. Comment on how these metrics provide insight into the model's performance.



True Positives (TP): 472 instances were correctly classified as class 0.
False Negatives (FN): 73 instances were incorrectly classified as class 0 when they actually belonged to class 1.
False Positives (FP): 2 instances were incorrectly classified as class 1 when they actually belonged to class 0.
True Negatives (TN): 1 instance was correctly classified as class 1.

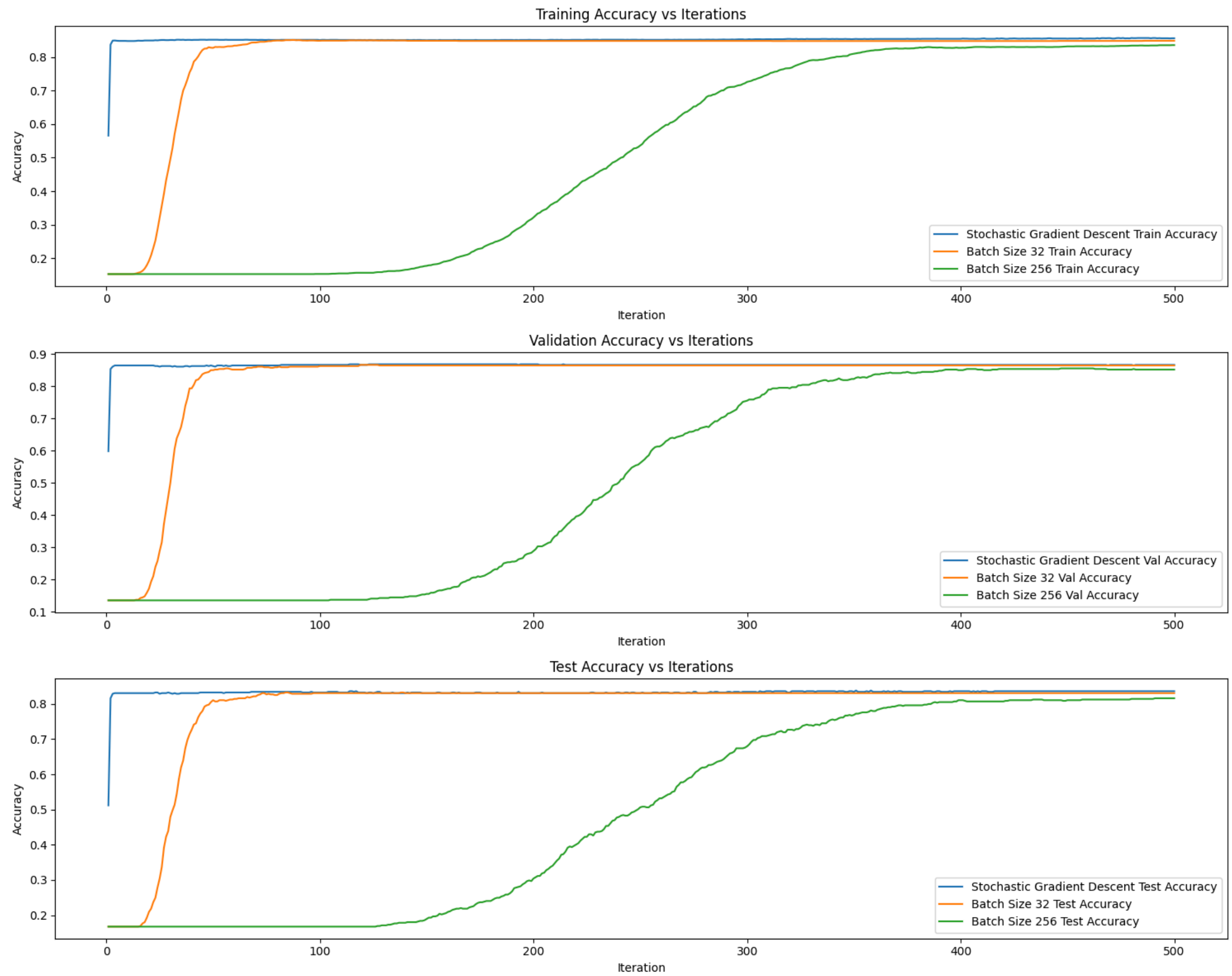
Observations:

- **Low Recall:** The very low recall score (0.0135) indicates that the model is struggling to identify most of the positive instances. This suggests that many positive instances are being misclassified as negative.
- **Moderate Precision:** The moderate precision score (0.333) suggests that when the model does predict a positive instance, there's a reasonable chance that it's correct. However, this doesn't offset the low recall.
- **Low F1-score:** The low F1-score (0.026) reflects the imbalance between precision and recall. A low F1-score indicates that the model is not performing well overall, likely due to the low recall.
- **Moderate ROC-AUC Score:** The ROC-AUC score of 0.745 is moderate. While it's not exceptionally high, it suggests that the model can reasonably differentiate between positive and negative instances to some extent.

Overall Assessment:

Based on these metrics, the model is performing poorly overall. The primary issue seems to be its inability to identify positive instances (low recall). While the precision is moderate, it doesn't compensate for the low recall. The moderate ROC-AUC score indicates that the model has some discriminatory power, but its overall performance is still unsatisfactory.

(d) (3 marks) Implement and compare the following optimization algorithms: Stochastic Gradient Descent and Mini-Batch Gradient Descent (with varying batch sizes, at least 2). Plot and compare the loss vs. iteration and accuracy vs. iteration for each method. Discuss the trade-offs in terms of convergence speed and stability between these methods.

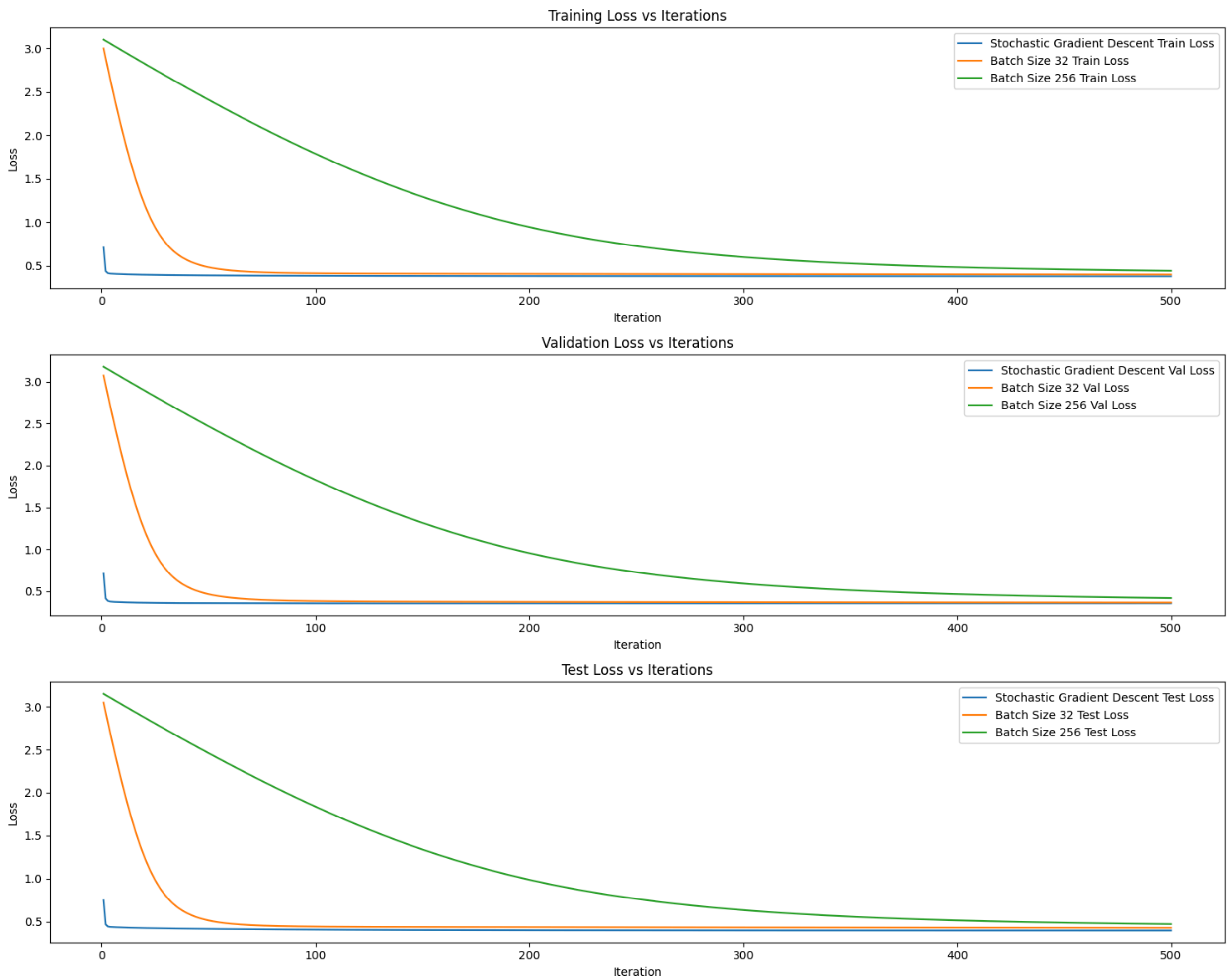


Observations

1. **Batch Size Impact on Training Speed:**
    - **Stochastic Gradient Descent (Batch Size 1):** The training accuracy fluctuates significantly, indicating noisy updates. This is because each data point is used to update the model's parameters individually.
    - **Batch Size 32:** The training accuracy converges more smoothly than Stochastic Gradient Descent, but still exhibits some fluctuations.
    - **Batch Size 256:** The training accuracy converges the most smoothly and quickly, suggesting that larger batch sizes can lead to faster convergence.
2. **Batch Size Impact on Generalization:**
    - **Validation and Test Accuracy:** For all batch sizes, the validation and test accuracy initially increase but eventually plateau. This indicates that the model is starting to overfit.
    - **Batch Size 32:** In this case, Batch Size 32 appears to have a slight edge in terms of validation and test accuracy compared to the other batch sizes, suggesting that it might be the best choice for this particular model and dataset.

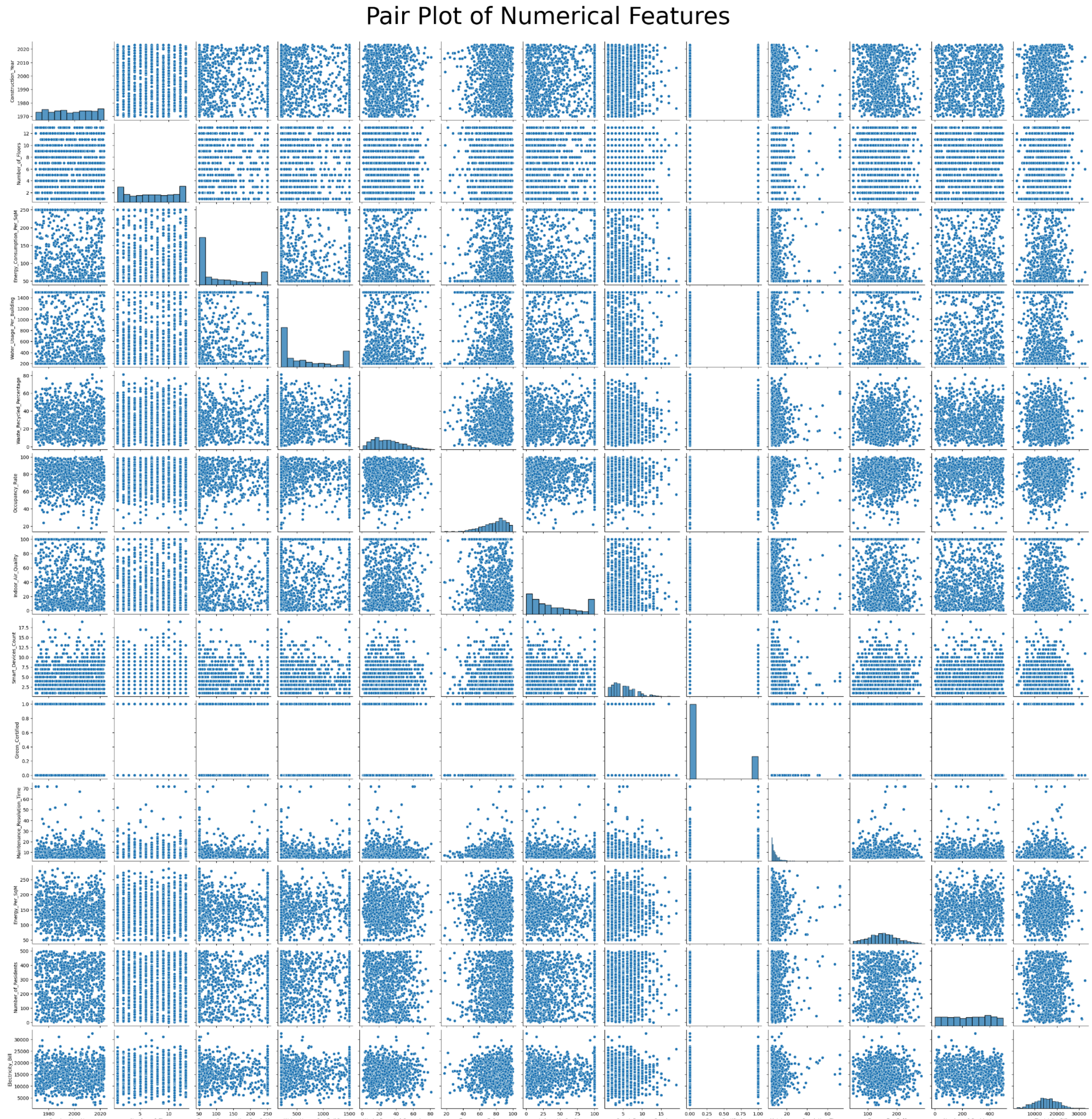
Conclusions

- **Larger batch sizes can lead to faster convergence.**
- **The choice of batch size can impact generalization performance.**



The choice of batch size significantly influences the training process and generalization performance of a machine learning model. Larger batch sizes can accelerate the training process, leading to quicker convergence. However, the impact on generalization, as measured by validation and test loss, varies. While larger batch sizes might speed up training, they can also contribute to overfitting, especially in certain scenarios. In this specific case, a batch size of 32 seems to strike a good balance between training speed and generalization performance, suggesting that it might be the optimal choice for this particular model and dataset.

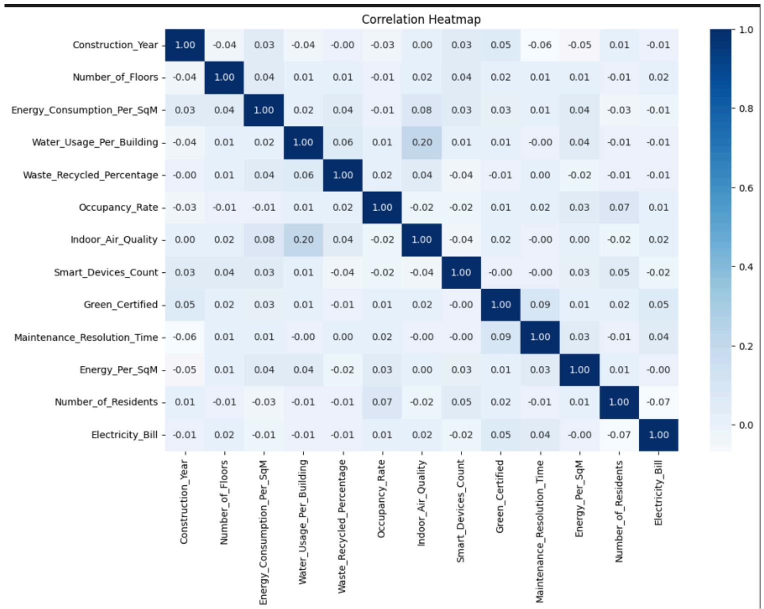
Section C



a) Perform EDA by creating pair plots, box plots, violin plots, count plots for categorical features, and a correlation heatmap. Based on these visualizations, provide at least five insights on the dataset.

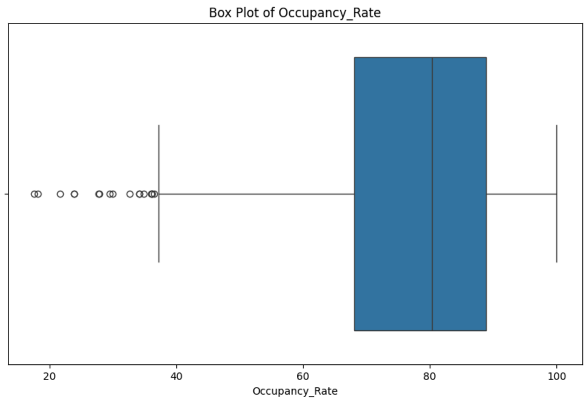
1. The correlation analysis reveals several significant relationships between the variables. Energy consumption is strongly linked to electricity bills, while water usage may

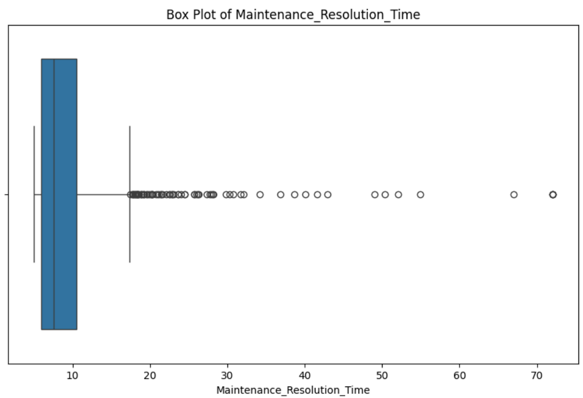
positively influence indoor air quality.



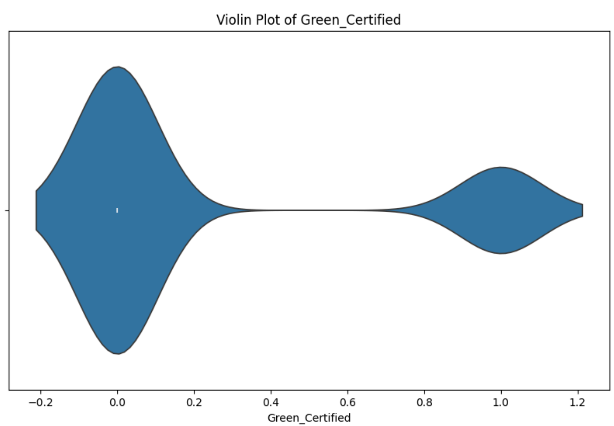
2. Construction year and the presence of smart devices do not seem to have a substantial impact on the factors under consideration. Buildings with higher occupancy rates tend to have higher waste recycling percentages, and faster maintenance resolution times are often associated with green certification.

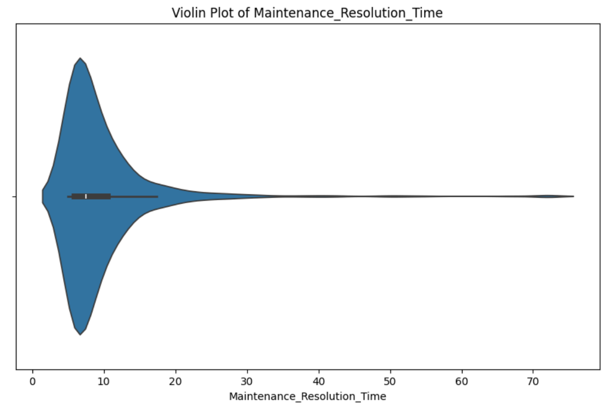
3. From the box plots of the variables Occupancy_Rate, Smart_Devices_Count, Maintenance_Resolution_Time, Energy_Per_SqM, and Electricity_Bill, it is evident that there are some outliers present. Notably, there are a **significant number of outliers in** the **Occupancy_Rate** and **Maintenance_Resolution_Time** variables.





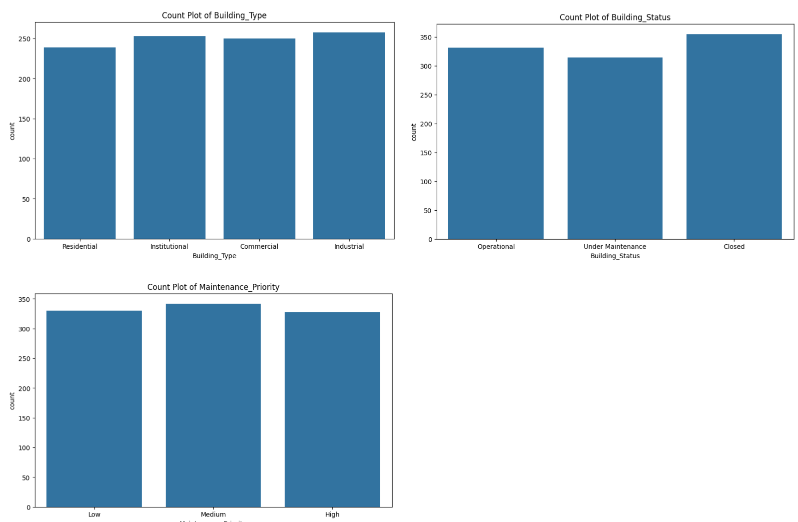
4. The violin plot for the **Green_Certified** variable indicates that the majority of samples are not green certified.



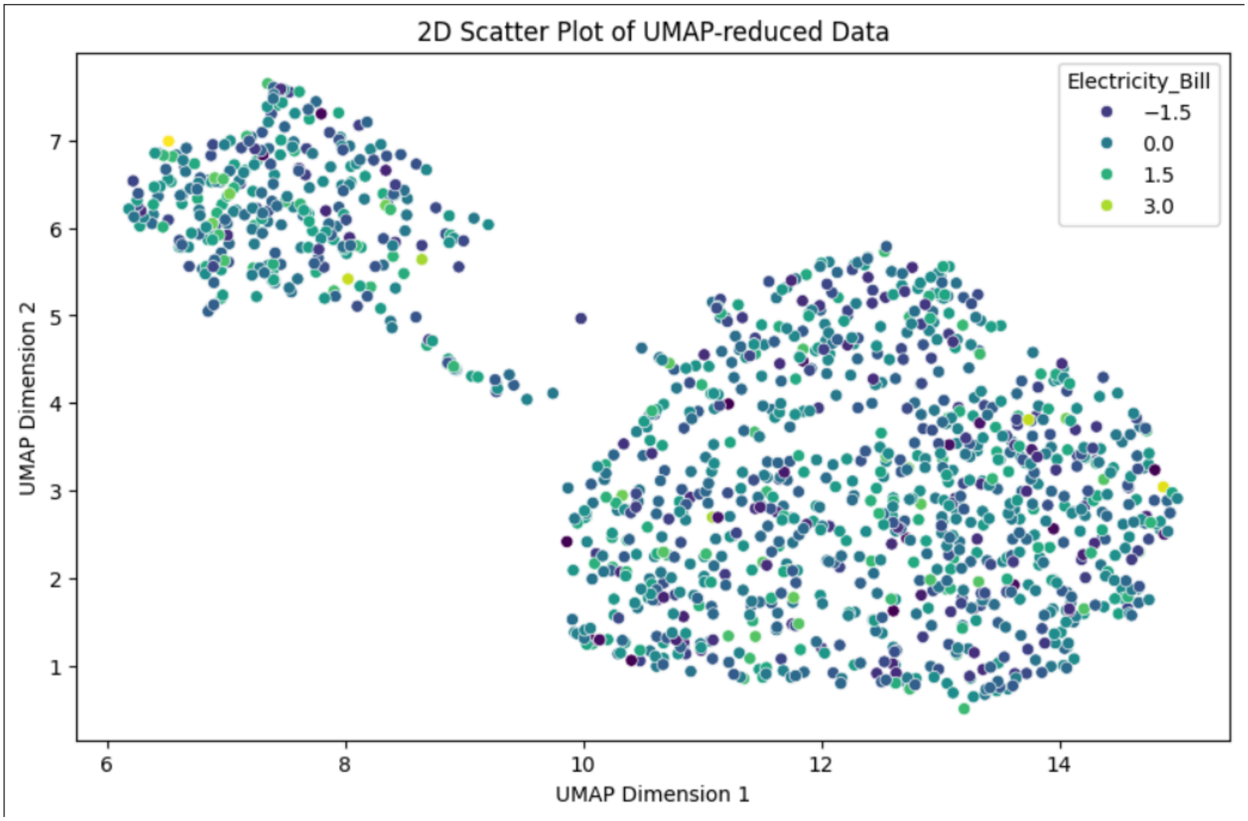


The violin plot of maintenance resolution time indicates that the majority of data points fall within the range of 0 to 25.

5. Based on the count plot, it can be concluded that the majority of buildings in the dataset are of industrial type, have a closed status, and are assigned a medium maintenance priority.



b) Comment on the separability and clustering of the data after dimensionality reduction.



UMAP dimensionality reduction has effectively maintained the core structure of the data and highlighted two distinct clusters. However, the minor overlap between clusters indicates that incorporating additional features or employing more advanced techniques could improve separation. Utilizing one-hot encoding instead of label encoding might yield better results.

c) Report Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R2 score, Adjusted R2 score,
and Mean Absolute Error (MAE) on the train and test data.

Train Data Metrics:
MSE: 24475013.1685
RMSE: 4947.2228
MAE: 4006.3285
R^2: 0.0139
Adjusted R^2: -0.0011

Test Data Metrics:
MSE: 24278016.1557
RMSE: 4927.2727
MAE: 3842.4093
R^2: 0.0000
Adjusted R^2: -0.0641

Based on these metrics, it appears that the model is not able to effectively predict the target variable. A **negative adjusted R^2** typically indicates that the model is performing **worse than a simple mean-based** model. This implies that **features that are not contributing positively to the model need to be removed.**

d) Train the regression model using the selected features. Compare the results (MSE, RMSE, R2 score, Adjusted R2 score, MAE) on the train and test dataset with the results obtained in part (c).

Selected Features: Index(**['Building_Type', 'Green_Certified', 'Number_of_Residents']**, dtype='object')
Train Data Metrics:
  MSE: 24569032.9069
  RMSE: 4956.7159
  MAE: 4006.4734
  R^2: 0.0101
  Adjusted R^2: 0.0072

Test Data Metrics:
  MSE: 23941409.0630
  RMSE: 4892.9959
  MAE: 3813.9481
  R^2: 0.0139
  Adjusted R^2: 0.0019

After applying **Recursive Feature Elimination (RFE)**, we observed an **improvement in the adjusted R^2** score. This implies that the selection of the most important features has led to a model that better explains the variance in the data, particularly with fewer features. The adjusted R^2 is a metric that penalizes the model for including irrelevant or excessive predictors, and the increase suggests that the model is now using a more relevant subset of features, leading to better generalization.

e) Encode the categorical features of the original dataset using One-Hot Encoding and perform Ridge Regression on the preprocessed data. Report the evaluation metrics (MSE, RMSE, R2 score, Adjusted R2 score, MAE). Compare the results with those obtained in part (c).

Train Data Metrics:
  MSE: 24188990.4560
  RMSE: 4918.2304
  MAE: 3976.5520
  R^2: 0.0254
  Adjusted R^2: 0.0066

Test Data Metrics:

MSE: 24125368.4749
RMSE: 4911.7582
MAE: 3797.2657
R^2: 0.0063
Adjusted R^2: -0.0758

The combination of **Ridge regression** and **One-Hot Encoding** led to better performance by both reducing overfitting (via Ridge) and handling categorical variables more appropriately (via One-Hot Encoding). This resulted in improved metrics such as **lower MSE, RMSE, and MAE, and higher R^2 values.** Although the improvements are relatively modest, these techniques have made the model more stable and generalizable.

f) Perform Independent Component Analysis (ICA) on the one-hot encoded dataset and choose the appropriate number of components (try 4, 5, 6, and 8 components). Compare the results (MSE, RMSE, R2 score, Adjusted R2 score, MAE) on the train and test dataset.

Applying ICA with 4 components
Train Data Metrics:
MSE: 24592706.0132
RMSE: 4959.1033
MAE: 3978.6904
R^2: 0.0092
Adjusted R^2: 0.0052

Test Data Metrics:
MSE: 24253149.5828
RMSE: 4924.7487
MAE: 3804.8968
R^2: 0.0011
Adjusted R^2: -0.0152

Applying ICA with 5 components
Train Data Metrics:
MSE: 24590518.3028
RMSE: 4958.8828
MAE: 3979.3888
R^2: 0.0093
Adjusted R^2: 0.0043

Test Data Metrics:
MSE: 24281264.2516
RMSE: 4927.6023
MAE: 3807.5283
R^2: -0.0001

Adjusted R^2: -0.0206

Applying ICA with 6 components
Train Data Metrics:
MSE: 24559819.1006
RMSE: 4955.7864
MAE: 3978.4366
R^2: 0.0105
Adjusted R^2: 0.0045

Test Data Metrics:
MSE: 24178745.7164
RMSE: 4917.1888
MAE: 3799.7922
R^2: 0.0041
Adjusted R^2: -0.0205

Applying ICA with 8 components
Train Data Metrics:
MSE: 24426776.1803
RMSE: 4942.3452
MAE: 3973.2845
R^2: 0.0159
Adjusted R^2: 0.0079

Test Data Metrics:
MSE: 24207624.1905
RMSE: 4920.1244
MAE: 3796.0912
R^2: 0.0029
Adjusted R^2: -0.0302

Train Data Metrics:

- **MSE, RMSE, and MAE** slightly improved as the number of components increased from 4 to 8. The **lowest values for these metrics are with 8 components**.
- **R^2** and **Adjusted R^2** values also improved as the number of components increased, suggesting better model fit on the training set with more components. **Here too metric with 8 components show worse performance.**

Test Data Metrics:

- The **Test MSE, RMSE, and MAE** values are slightly lower for **6 components** compared to other component settings. However, the performance doesn't consistently improve as the number of components increases.

- **R^2** and **Adjusted R^2** values show small variations, with the **best R^2** for the test set occurring with **6 components** ($R^2 = 0.0041$), but the test performance remains quite modest overall.

Although adding more components through ICA improved training metrics, it did not necessarily translate to better test performance. **ICA with 6 components** seems to offer the most balanced results, but overall, the gains from applying ICA are modest.

(g) Use ElasticNet regularization (which combines L1 and L2) while training a linear model on the preprocessed dataset from part (c). Compare the evaluation metrics (MSE, RMSE, R2 score, Adjusted R2 score, MAE) on the test dataset for different values of the mixing parameter (alpha).

The ElasticNet regularization was applied with **alpha** values ranging from 0.01 to 10, and the performance metrics were analyzed for both the **train** and **test** datasets.

alpha: 0.01
Train Data Metrics:
  MSE: 24475021.7061
  RMSE: 4947.2236
  MAE: 4006.2253
  R^2: 0.0139
  Adjusted R^2: -0.0011

Test Data Metrics:
  MSE: 24276757.8815
  RMSE: 4927.1450
  MAE: 3842.2209
  R^2: 0.0001
  Adjusted R^2: -0.0640

alpha: 0.1
Train Data Metrics:
  MSE: 24475793.2838
  RMSE: 4947.3016
  MAE: 4005.3930
  R^2: 0.0139
  Adjusted R^2: -0.0011

Test Data Metrics:
  MSE: 24266701.4289
  RMSE: 4926.1244
  MAE: 3841.0938

R^2: 0.0005
Adjusted R^2: -0.0636

alpha: 1
Train Data Metrics:
MSE: 24512862.9956
RMSE: 4951.0467
MAE: 4001.7690
R^2: 0.0124
Adjusted R^2: -0.0027

Test Data Metrics:
MSE: 24231900.6448
RMSE: 4922.5908
MAE: 3835.0163
R^2: 0.0019
Adjusted R^2: -0.0620

alpha: 10
Train Data Metrics:
MSE: 24714505.1994
RMSE: 4971.3685
MAE: 4004.7030
R^2: 0.0043
Adjusted R^2: -0.0109

Test Data Metrics:
MSE: 24303481.5553
RMSE: 4929.8561
MAE: 3837.4646
R^2: -0.0010
Adjusted R^2: -0.0652

Conclusion:

- **Alpha = 0.01 and 0.1** provide the best balance between fitting the data well and controlling overfitting, as shown by the better metrics across both the train and test sets.
- **Higher Alpha values** lead to worse performance, especially on the training set, as the model becomes too constrained.
- ElasticNet with a low alpha value effectively balances between L1 and L2 regularization, leading to better overall model performance compared to larger alpha values.

(h) Use the Gradient Boosting Regressor to perform regression on the preprocessed dataset from part (c). Report the evaluation metrics (MSE, RMSE, R2 score, Adjusted R2 score, MAE). Compare the results with those obtained in parts (c) and (g).

Train Data Metrics:
MSE: 14926446.2573
RMSE: 3863.4759
MAE: 3092.7482
R^2: 0.3986
Adjusted R^2: 0.3895

Test Data Metrics:
MSE: 24392500.9011
RMSE: 4938.8765
MAE: 3815.7032
R^2: -0.0047
Adjusted R^2: -0.0691

1. **Train Data Performance**:
    - The **Gradient Boosting Regressor** significantly outperforms both **Ridge Regression** and **ElasticNet** on the training data, as seen by the much lower **MSE** (14.9M vs. ~24.5M), **RMSE**, and **MAE**.
    - The **R^2** score of **0.3986** is much higher for Gradient Boosting, showing that it captures more variance in the training data.
2. **Test Data Performance**:
    - Despite the excellent training performance, the **Gradient Boosting Regressor** shows poorer generalization to the test data with a negative **R^2** (-0.0047) and higher **MSE** and **RMSE** compared to the **Ridge Regression** and **ElasticNet** models.
    - The **Gradient Boosting** model shows signs of **overfitting**. This is evident from the large discrepancy between the train and test performance, as the model fits the training data too well but generalizes poorly to new data.
3. **ElasticNet vs. Ridge**:
    - **ElasticNet** (alpha = 0.01) and **Ridge Regression** perform nearly identically in both training and test datasets, as expected with the close metrics across the board. However, **ElasticNet** has a marginally better **R^2** and **Adjusted R^2** on the test set, though the difference is minimal.
4. **Model Choice**:
    - If generalization to test data is a priority, **ElasticNet (alpha = 0.01)** or **Ridge Regression** would be preferable, as both maintain more stable performance across train and test datasets.

    - If the focus is solely on training performance and capturing complex patterns in the data, **Gradient Boosting** may be considered, but tuning is required to avoid overfitting.

Conclusion:

- **Gradient Boosting Regressor** excels in the training phase but suffers from overfitting, leading to poor test performance.
- **Ridge Regression** and **ElasticNet** (alpha = 0.01) provide more consistent and reliable results across both training and test datasets, making them better choices for this problem.