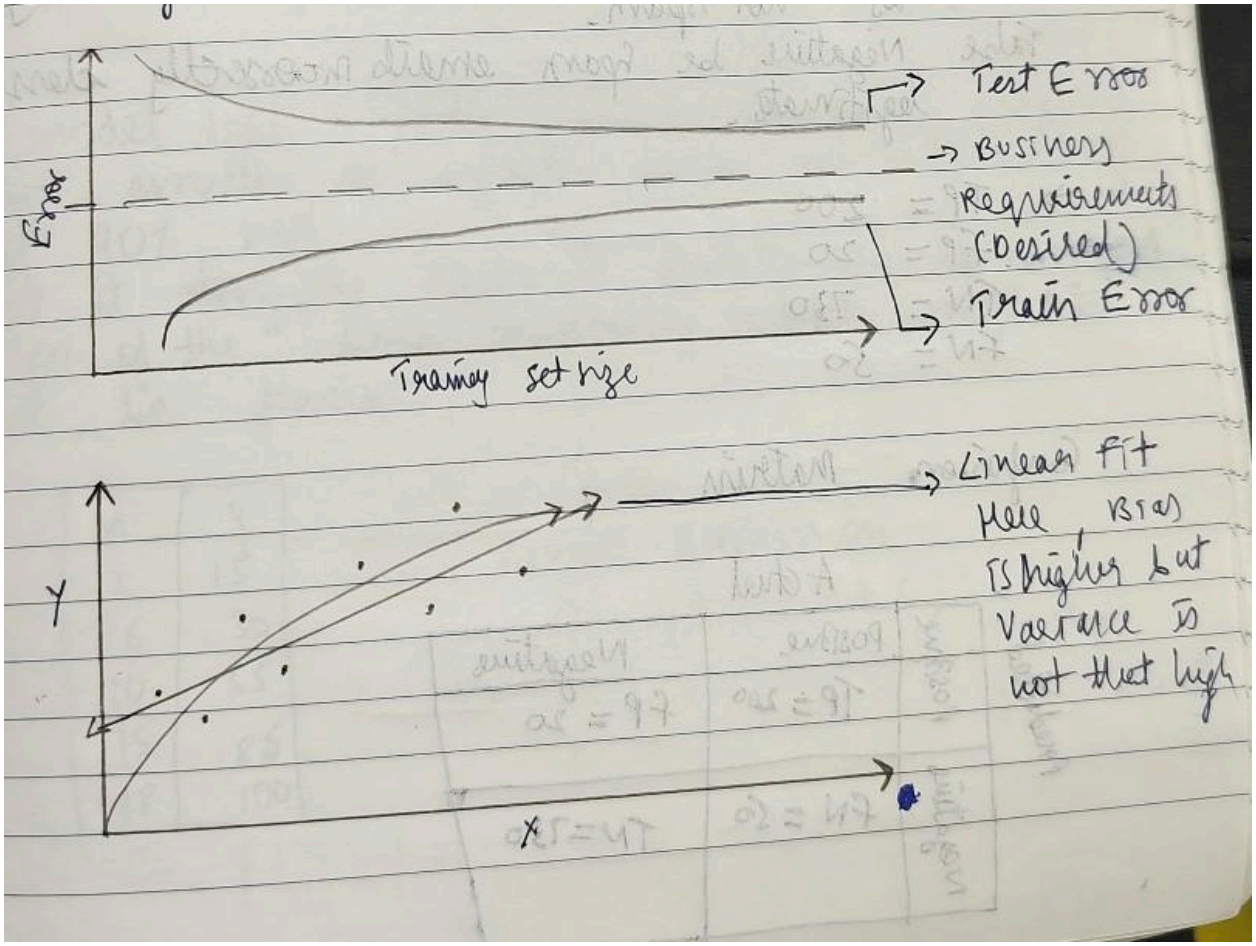
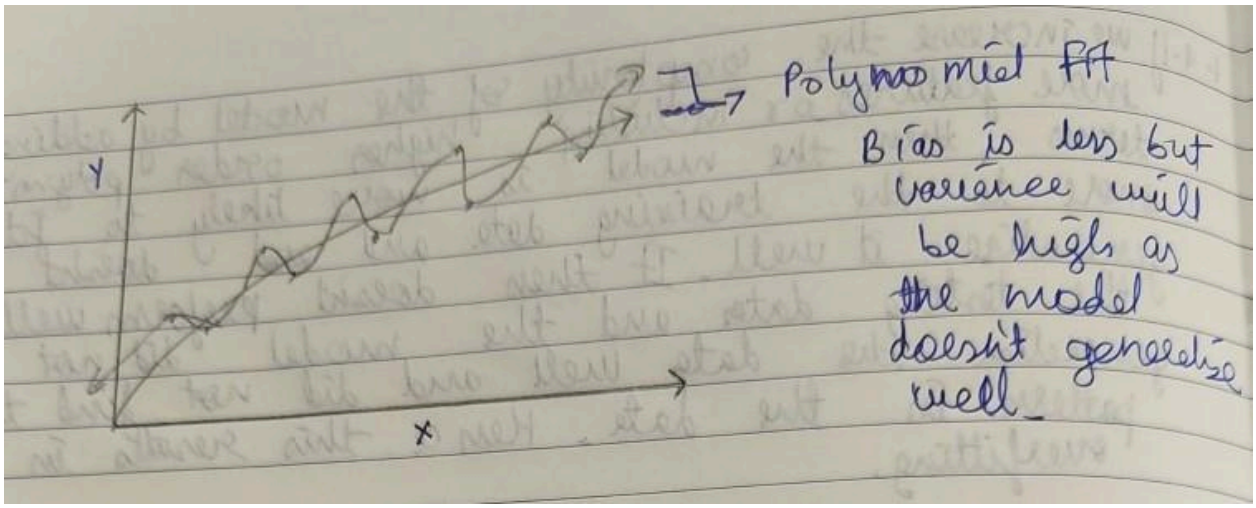
ML Assignment 1

Vikranth Udandarao
2022570

1. a. If we increase the complexity of the model by adding more features or including higher-order polynomial terms, then the model is more likely to fit the training data and not generalize it well. It then doesn't perform well on the testing data and the model is overfitting. The model did not generalize the data well and did not just fit the patterns in the data. Hence, this results in overfitting.

Bias decreases as model performs really well on training data as it catches more restricted patterns but variance increases as the model will not likely perform well on testing data as it has not generalized the data well.





b. Let:

- **True Positive (TP)** be: *Emails correctly identified as spam*.
- **False Positive (FP)** be: *Emails incorrectly identified as spam but actually legitimate*.
- **True Negative (TN)** be: *Legitimate emails correctly identified as not spam*.
- **False Negative (FN)** be: *Spam emails incorrectly classified as legitimate*.

Values:

- TP = 200
- FP = 20
- TN = 730
- FN = 50

Confusion Matrix:

	Actual Positive	**Actual Negative**
Predicted Positive	TP = 200	FP = 20
Predicted Negative	FN = 50	TN = 730

Precision = TP / (TP + FP)
= 200 / (200 + 20)
= 200 / 220
= 0.909
= 90.9%

Recall = TP / (TP + FN)
= 200 / (200 + 50)
= 200 / 250
= 0.8
= 80%

F1 Score = 2 × (Precision × Recall) / (Precision + Recall)
= 2 × (0.909 × 0.8) / (0.909 + 0.8)
= 2 × 0.727 / 1.709
= 0.851
= 85.1%

Accuracy = (TP + TN) / (TP + FP + TN + FN)
= (200 + 730) / (200 + 20 + 730 + 50)
= 930 / 1000

= 0.93
= 93%

Conclusion:
The model does a good job in accurately flagging spam emails as spam with an accuracy of 90%. But it classifies legitimate emails as spam about 2.67% of the time incorrectly, which might not be feasible.

c. **Linear Regression**:
$y=m \times x + c$,
m = slope,
c = intercept.

$$m = \frac{N\Sigma xy - \Sigma x \, \Sigma y}{N\Sigma x^2 - (\Sigma x)^2}$$

x	y	xy	x^2
3	15	45	9
6	30	180	36
10	55	550	100
15	85	1275	225
18	100	1800	324

$\Sigma x = 52$, $\Sigma y = 285$, $\Sigma xy = 3850$, $\Sigma x^2 = 694$.

$$m = \frac{5(3850) - (52)(285)}{5(694) - (52)^2}$$

$$= \frac{19250 - 14820}{3470 - 2704}$$

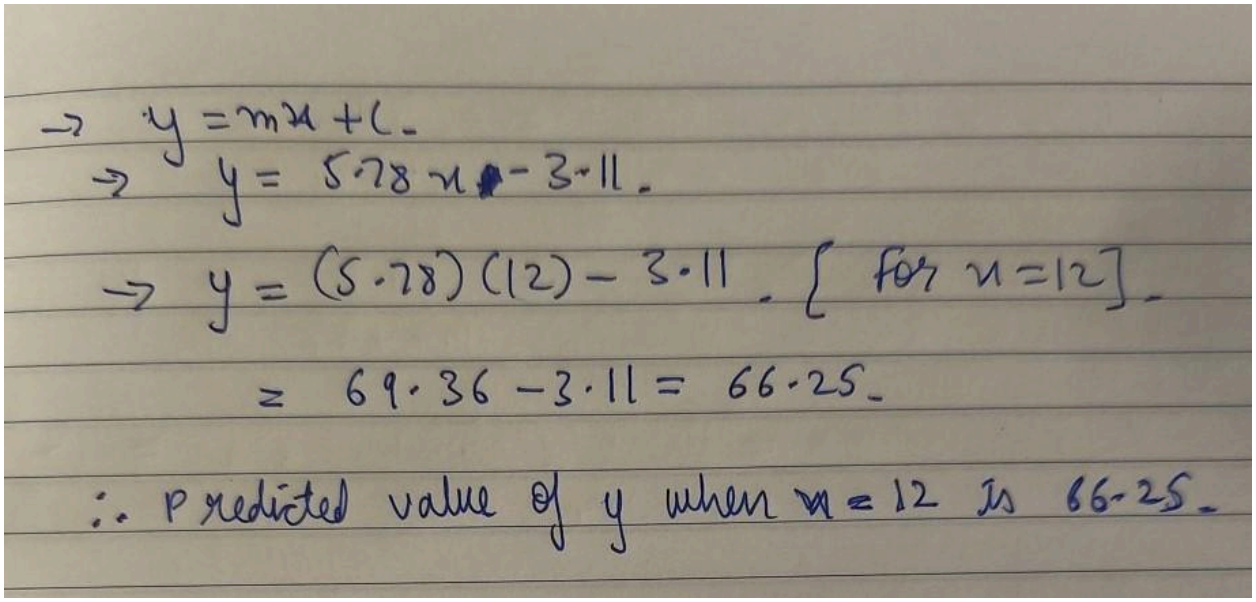
$$= \frac{4430}{766} \approx 5.78.$$

$\rightarrow m = 5.78$

$$c = \frac{\Sigma y - m(\Sigma x)}{N}$$

$$= \frac{285 - (5.78)(52)}{5}$$

$$= \frac{285 - 300.56}{5} = \left|\frac{-15.56}{5}\right| \approx -3.11.$$



d. To demonstrate a situation where a model f1 has a lower empirical risk (training error) compared to another model f2 but may not generalize better, we can construct a simple toy example using overfitting models.

Assume X={1,2,3,4,5},Y={12,22,32,42,52}={1,4,9,16,25}

This data follows the quadratic function Y=X^2.

Model f1: High-degree Polynomial (Overfitting Model)

- We fit a 4th-degree polynomial to the training data. A 4th-degree polynomial is flexible enough to fit the 5 training points exactly. For simplicity, assume that f1f_1f1 perfectly fits the points, which results in **0 training error**.

Model f2: Simple Linear Model (Model)

- We fit a linear model of the form f2(x)=ax+bf_2(x) = ax + bf2(x)=ax+b. This model cannot perfectly capture the quadratic nature of the data but can still approximate the relationship.

d- $X = \{1, 2, 3, 4, 5\}$

$Y = \{1^2, 2^2, 3^2, 4^2, 5^2\} = \{1, 4, 9, 16, 25\}$.

Models-

$f_1 \rightarrow$ 4 degree polynomial.

$f_2 \rightarrow$ Linear polynomial.

$f_2(x) = ax + b$.

$$a = \frac{N \Sigma xy - \Sigma x \Sigma y}{N \Sigma x^2 - (\Sigma x)^2}$$

$$b = \frac{\Sigma y - a \Sigma x}{N}$$

$$\rightarrow a = \frac{5 \times 225 - 15 \times 55}{5 \times 55 - 15^2} = 6.$$

$$b = \frac{55 - 6 \times 15}{5} = -7.$$

→ $y = 9x + 6 = 6x - 7$.

MSE:

$f_2(1) = 6(1) - 7 = -1$
$f_2(2) = 6(2) - 7 = 5$
$f_2(3) = 6(3) - 7 = 11$
$f_2(4) = 6(4) - 7 = 17$
$f_2(5) = 6(5) - 7 = 23$
~~$f_2(6) = 6(6) - 7$~~

Error:
$E(1) = 4$
$E(2) = 1$
$E(3) = 4$
$E(4) = 1$
$E(5) = 4$

→ $MSE_{f_2} = \frac{1}{5}[4 + 1 + 4 + 1 + 4] = \frac{14}{5} = 2.8$

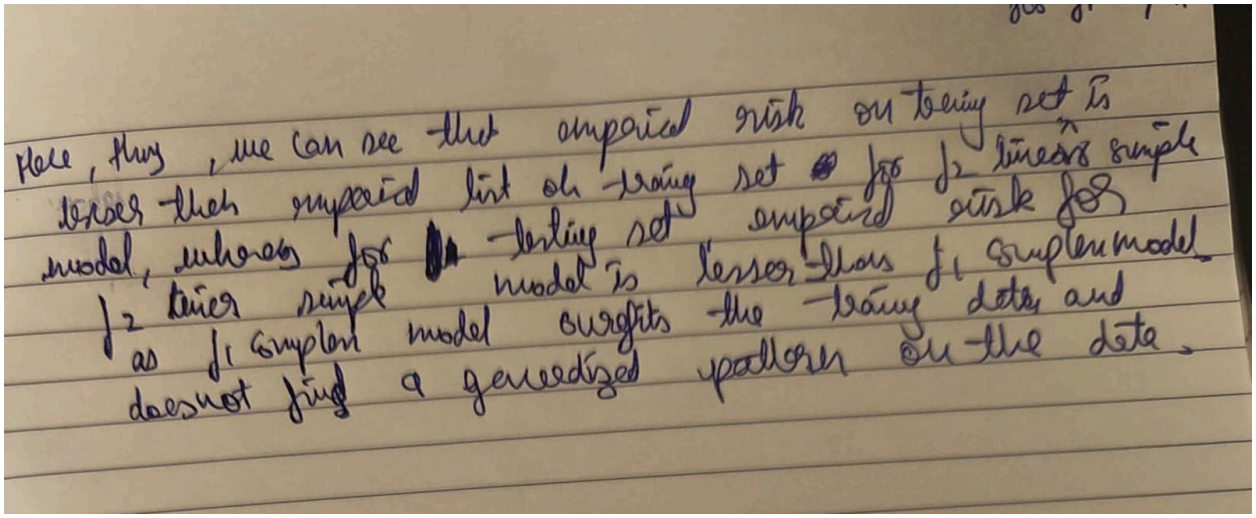
Suppose, $x_{test} = (6^2, 7^2) = (36, 49)$.

$f_1(6) = 28$
$f_{2+}(7) = 35$ - [As f_1 will differ as complex model]

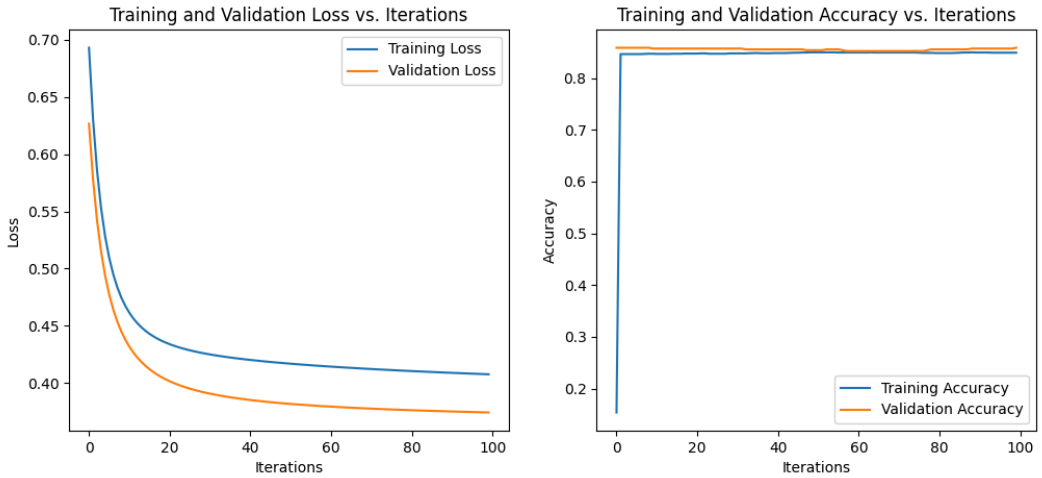
But for f_2,

$f_2(6) = 29$
$f_2(7) = 35$

→ $MSE_{test} = \frac{(36-29)^2 + (49-35)^2}{2} = 122.5$



2. a.



Learning Rate = 0.1
Iterations = 100

1. **Training and Validation Loss vs. Iterations**
    - The training and validation loss curves steadily decline, indicating that the model is learning effectively.
    - The validation loss decreases rapidly in the early iterations but stabilizes at a lower value than the training loss. This could indicate that the model generalizes well to the validation set. The training loss is slightly higher, which may suggest the presence of some regularization, or it could mean that the training data is more complex.
    - Both curves eventually flatten out, showing that the model is converging. The decreasing slope becomes negligible around iteration 80, suggesting that additional iterations are unlikely to improve the model's performance significantly.
2. **Training and Validation Accuracy vs. Iterations**

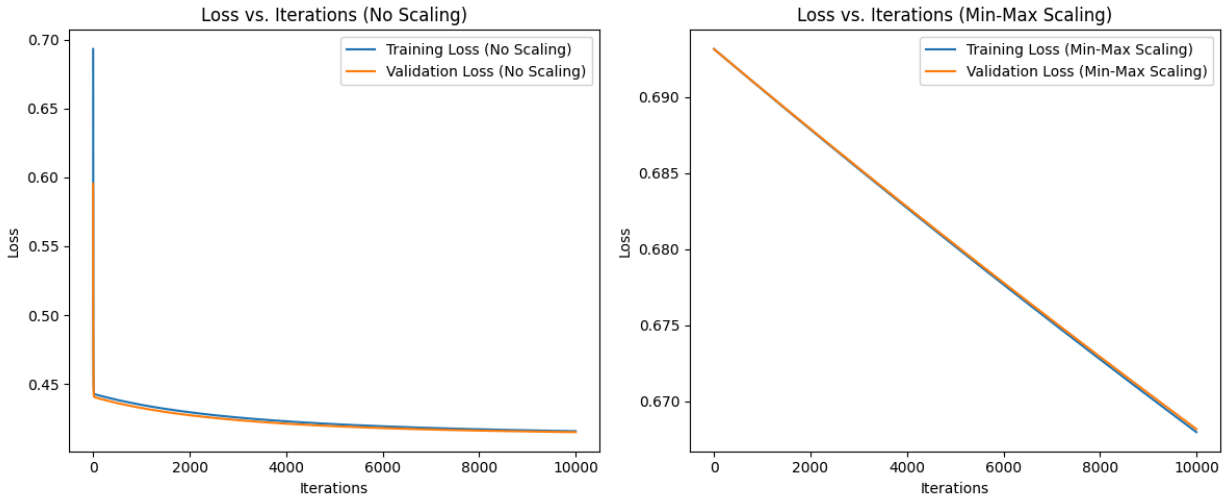
- The accuracy for both training and validation quickly reaches a high value (above 80%) within the first few iterations.
- Both accuracies remain almost identical and stable throughout the remaining iterations, which suggests that the model is not overfitting or underfitting. This high and consistent accuracy indicates good model performance and generalization across training and validation sets.

The model converges efficiently, as seen from the flattening of both the loss and accuracy curves. This suggests that the optimization process (gradient descent) is functioning well and that the learning rate is appropriate. No signs of instability or divergence are evident in the loss curves.

The gap between training and validation loss is minimal, which suggests that the model is not overfitting. Similarly, the nearly identical training and validation accuracy curves confirm that the model generalizes well to unseen data.

The model's accuracy plateaus after very few iterations, indicating that it learns the decision boundary relatively quickly. However, the flattening of the loss curve at higher iterations shows that additional training does not offer significant improvements.

b.



Learning Rate = 0.00001
Iterations = 10000

1. Performance of Model Without Scaling

- The model starts with a relatively high training and validation loss of approximately **0.70**.
- The loss decreases rapidly within the first few iterations, settling around **0.43** after about **10,000 iterations**. However, the convergence is slow, as can be observed from the steady yet prolonged reduction in loss.

- The model shows a slow convergence when no feature scaling is applied. It takes a significant number of iterations to reach the final loss values, indicating that the gradient descent is not optimized for feature ranges without scaling.

2. Performance of Model With Min-Max Scaling

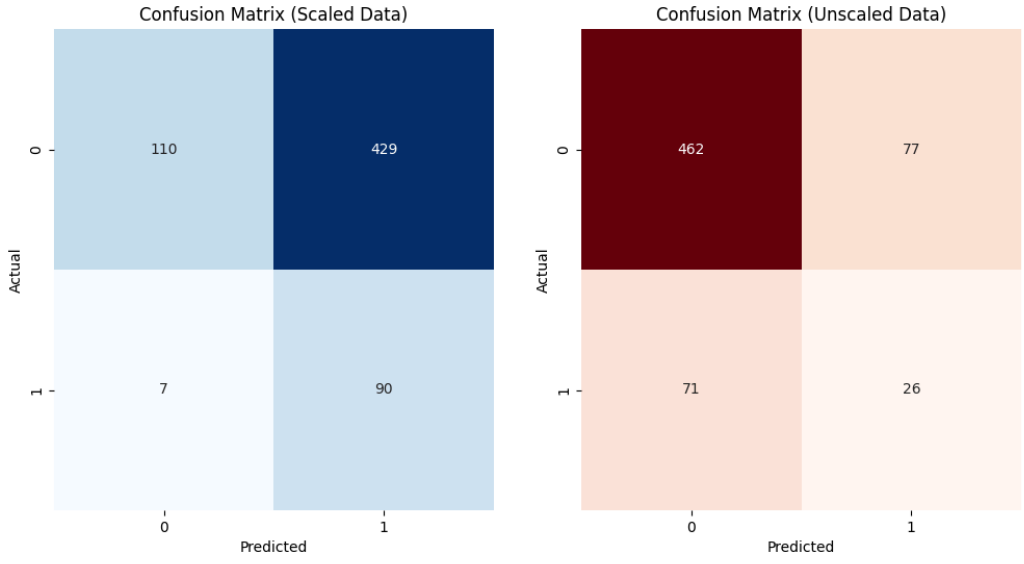
- The model demonstrates the impact of Min-Max scaling, where both the training and validation losses start lower, around **0.69**, and decrease steadily.
- Within a few thousand iterations, the model is already showing significant improvement, and the losses reduce consistently, achieving a final loss of **0.684** at around **10,000 iterations**.
- Min-Max scaling helps accelerate the convergence, as the loss reduces more smoothly over time. The difference between training and validation losses remains small throughout the process, indicating stable performance.

3. Impact of Feature Scaling on Model Convergence

- **Without Scaling:** The model shows slower convergence and requires more iterations to reduce the loss. The loss values are relatively higher during the initial stages of training, and the gradient descent struggles to make fast progress.
- **With Min-Max Scaling:** The model converges more quickly, and the loss decreases at a faster rate. This is likely because Min-Max scaling brings all features to a uniform range, allowing gradient descent to navigate the optimization landscape more efficiently.

c.

Confusion Matrix:



Learning Rate = 0.0001

Iterations = 10000

Min-Max Scaled:

- **Precision:** 0.1734
- **Recall:** 0.9278
- **F1 Score:** 0.2922
- **ROC-AUC Score:** 0.6472
- **Training Accuracy (Scaled):** 0.3125
- **Validation Accuracy (Scaled):** 0.3145

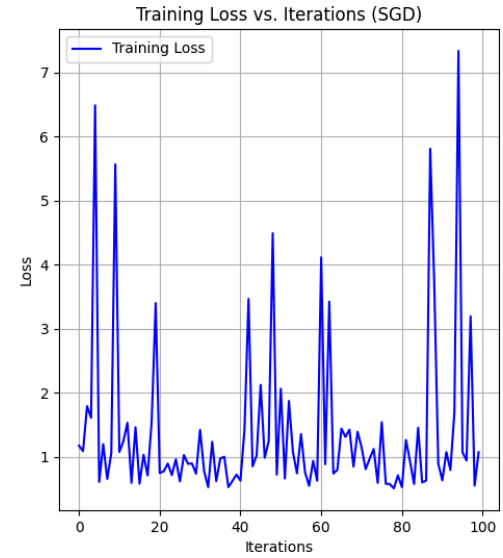
- **Precision:** The precision is low at **17.34%**, indicating that a significant proportion of the model's positive predictions are incorrect.
- **Recall:** The recall is high at **92.78%**, showing that the model is very sensitive in capturing true positives (classifying the positive class correctly).
- **F1 Score:** The F1 score of **29.22%** reflects the imbalance between precision and recall, suggesting that the model is prioritizing recall over precision.
- **ROC-AUC:** The ROC-AUC score of **64.72%** suggests that the model has a moderate ability to distinguish between the positive and negative classes, but there's room for improvement.

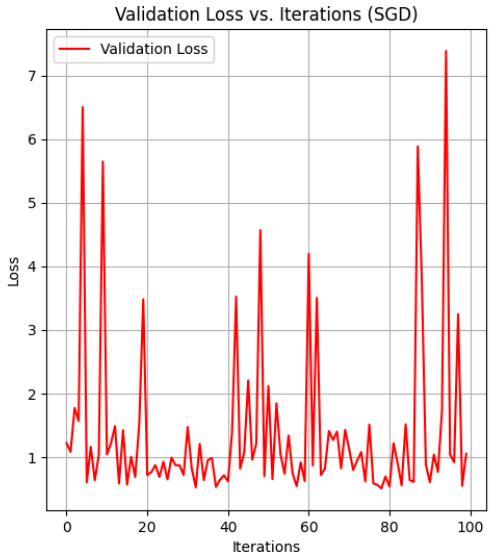
Unscaled:

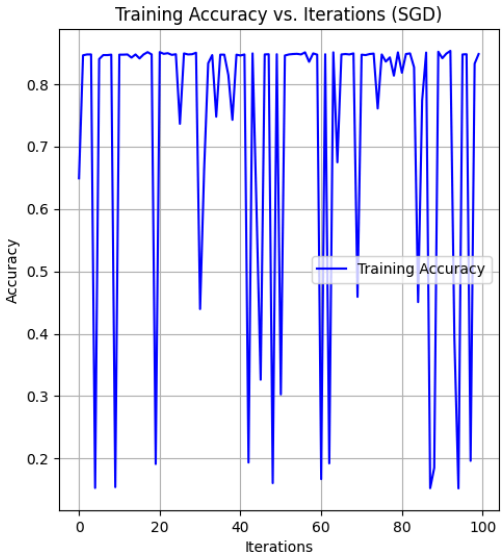
- **Precision:** 0.2524
- **Recall:** 0.2680
- **F1 Score:** 0.2600
- **ROC-AUC Score:** 0.5404
- **Training Accuracy (Unscaled):** 0.7404
- **Validation Accuracy (Unscaled):** 0.7673

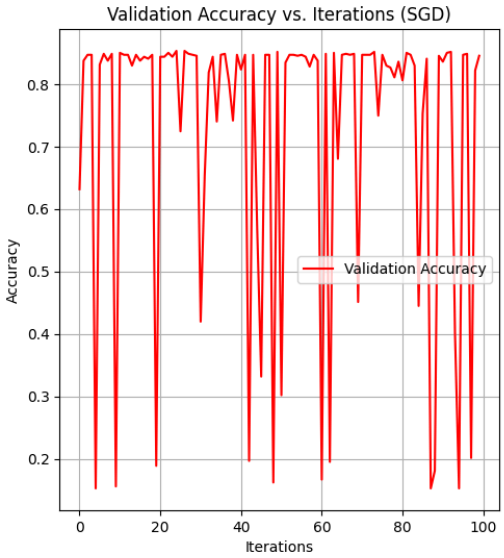
- **Precision:** The precision for unscaled data is higher than scaled data at **25.24%**, indicating that the model makes fewer false positive predictions.
- **Recall:** The recall is much lower than with scaled data at **26.80%**, showing that the model misses a substantial number of true positives.
- **F1 Score:** The F1 score of **26.00%** is lower than the scaled data, highlighting the trade-off between precision and recall, with a stronger bias toward improving precision at the cost of recall.
- **ROC-AUC:** The ROC-AUC score of **54.04%** shows the model has a limited ability to separate the two classes, with a near-random classification boundary.

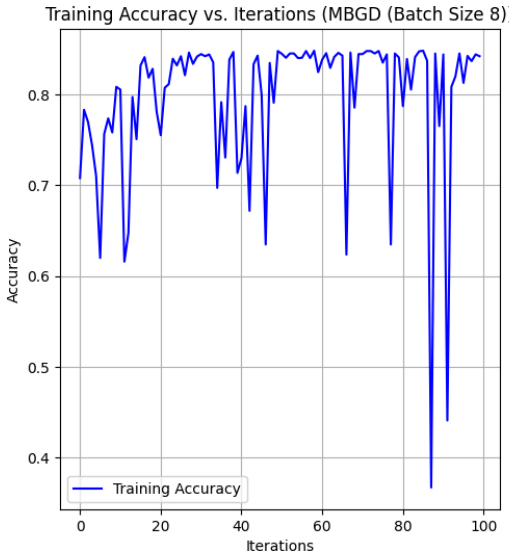
d.

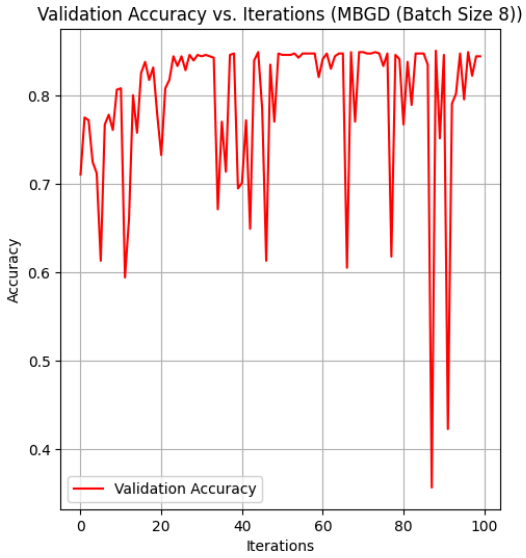

Training Loss vs. Iterations (SGD)
Training Loss
Loss
Iterations

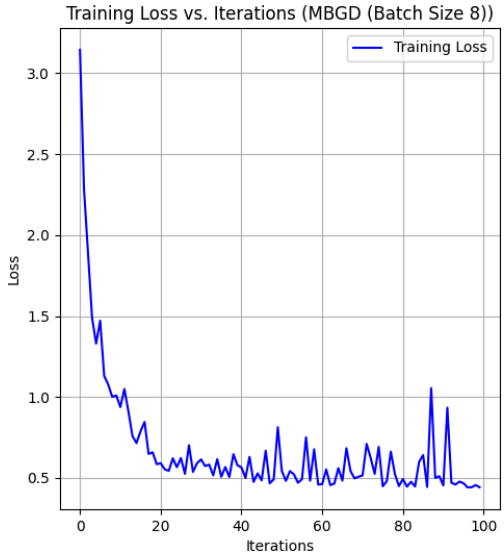

Validation Loss vs. Iterations (SGD)
Validation Loss
Loss
Iterations

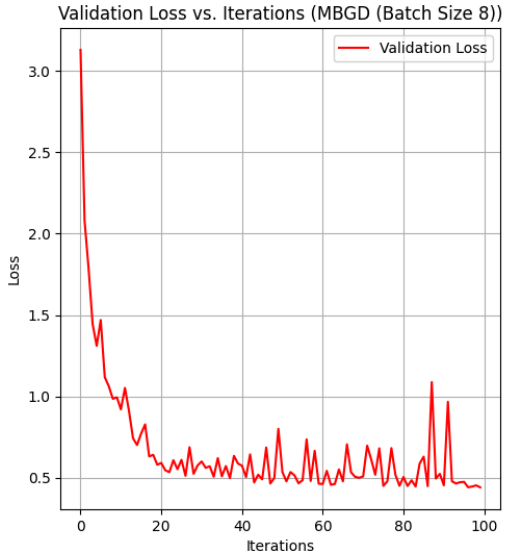

Training Accuracy vs. Iterations (SGD)
Training Accuracy
Accuracy
Iterations


Validation Accuracy vs. Iterations (SGD)
Validation Accuracy
Accuracy
Iterations

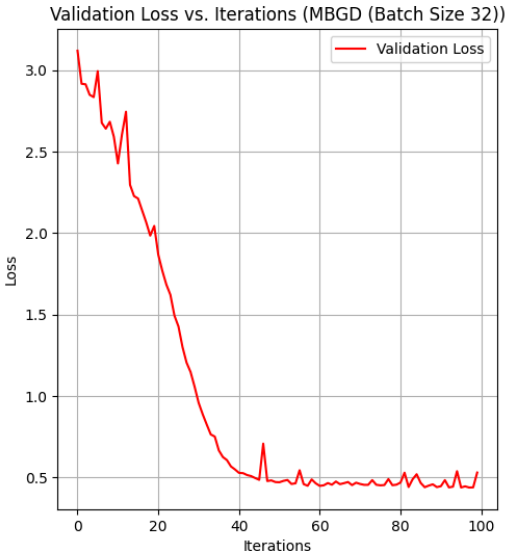

Training Accuracy vs. Iterations (MBGD (Batch Size 8))
Accuracy
Iterations
Training Accuracy
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

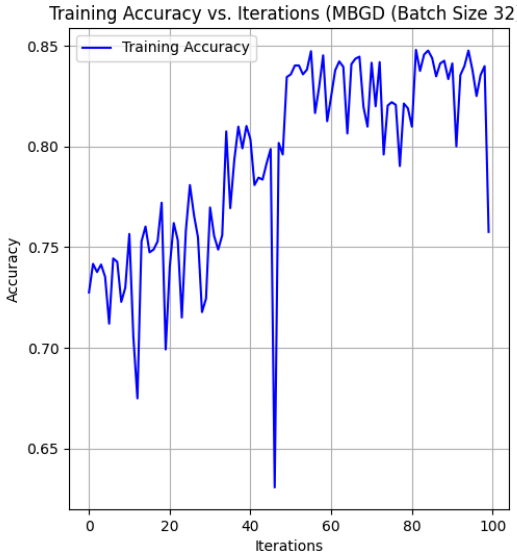

Validation Accuracy vs. Iterations (MBGD (Batch Size 8))
Accuracy
Iterations
Validation Accuracy
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

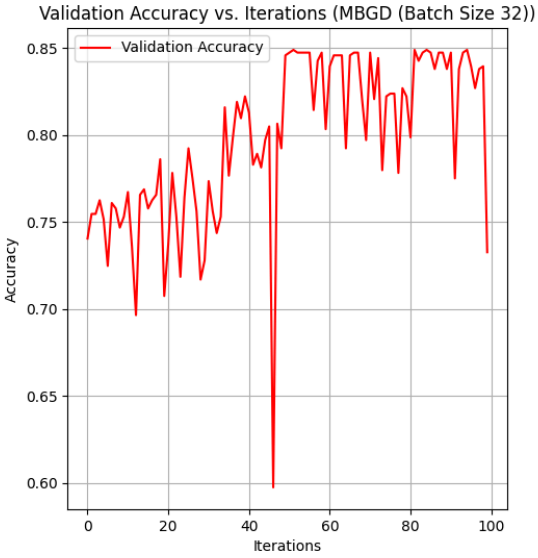

Training Loss vs. Iterations (MBGD (Batch Size 8))
Loss
Iterations
Training Loss
0.5
1.0
1.5
2.0
2.5
3.0
0
20
40
60
80
100

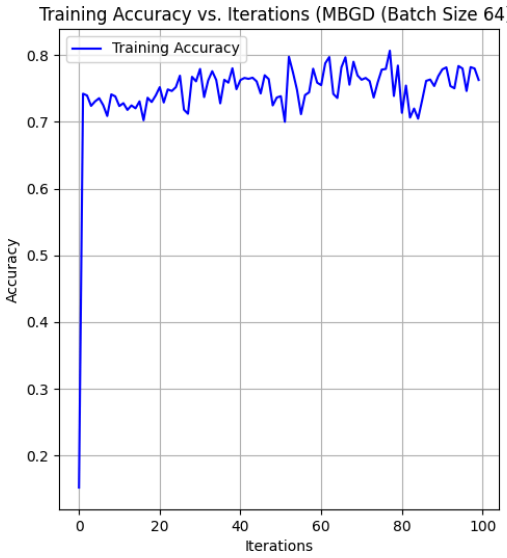

Validation Loss vs. Iterations (MBGD (Batch Size 8))
Loss
Iterations
Validation Loss
0.5
1.0
1.5
2.0
2.5
3.0
0
20
40
60
80
100

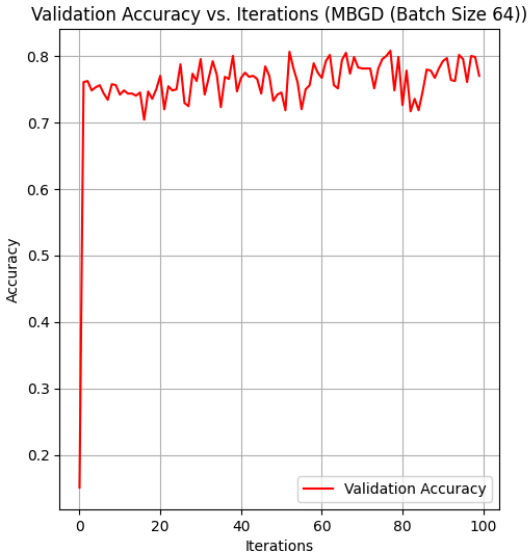

Training Loss vs. Iterations (MBGD (Batch Size 32))
Training Loss
Loss
Iterations

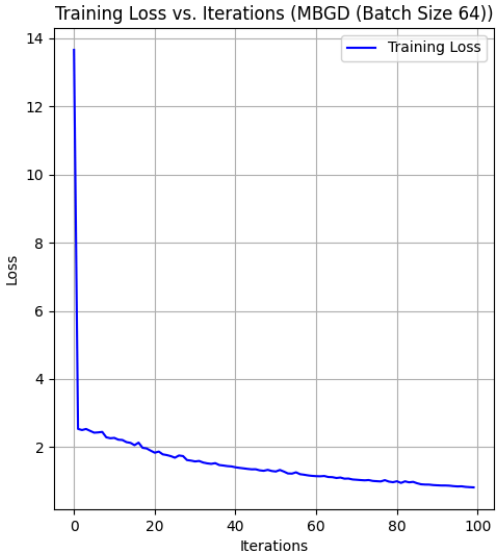

Validation Loss vs. Iterations (MBGD (Batch Size 32))
Validation Loss
Loss
Iterations

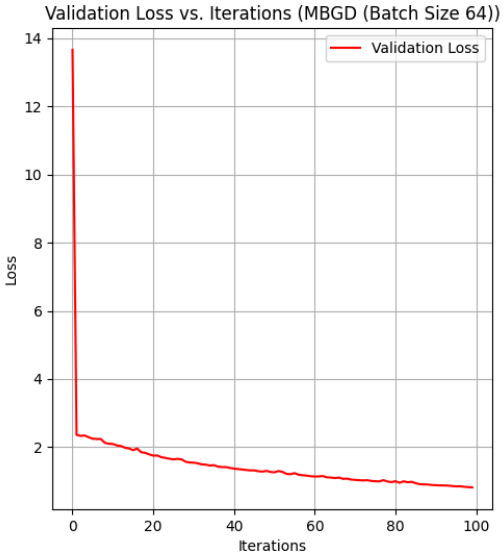

Training Accuracy vs. Iterations (MBGD (Batch Size 32))
Training Accuracy
Accuracy
Iterations

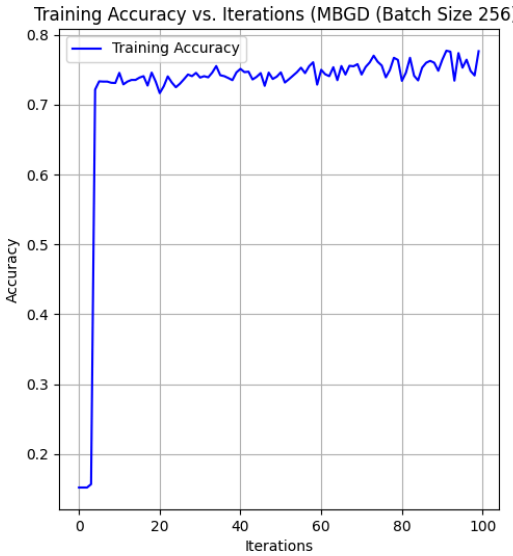

Validation Accuracy vs. Iterations (MBGD (Batch Size 32))
Validation Accuracy
Accuracy
Iterations

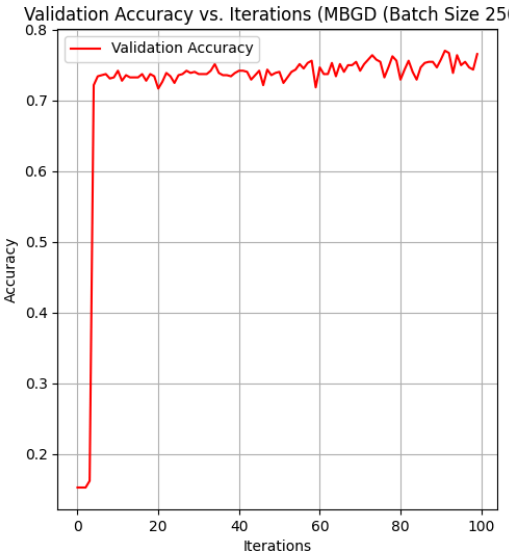

Training Accuracy vs. Iterations (MBGD (Batch Size 64))
Training Accuracy
Accuracy
Iterations
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

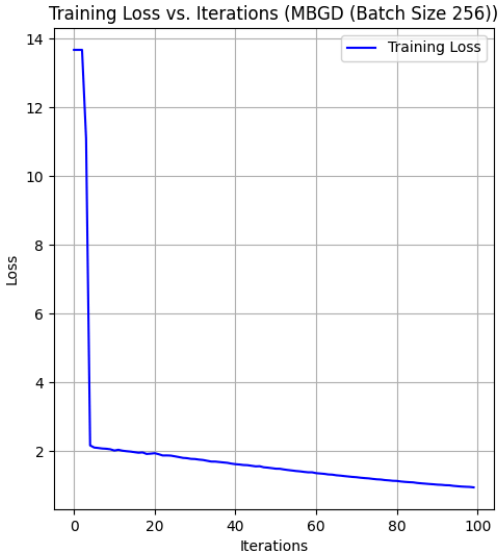

Validation Accuracy vs. Iterations (MBGD (Batch Size 64))
Validation Accuracy
Accuracy
Iterations
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

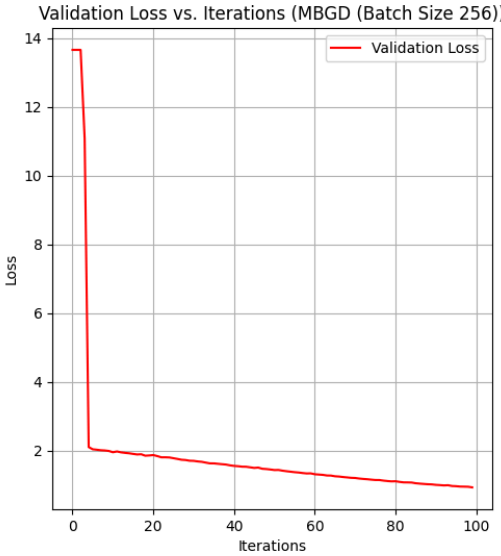

Training Loss vs. Iterations (MBGD (Batch Size 64))
Training Loss
Loss
Iterations
2
4
6
8
10
12
14
0
20
40
60
80
100

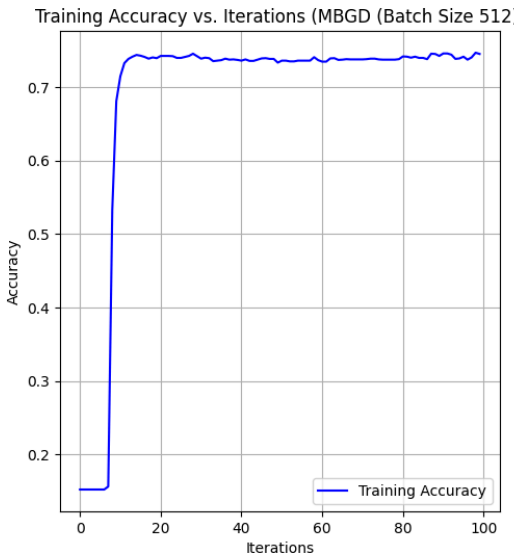

Validation Loss vs. Iterations (MBGD (Batch Size 64))
Validation Loss
Loss
Iterations
2
4
6
8
10
12
14
0
20
40
60
80
100

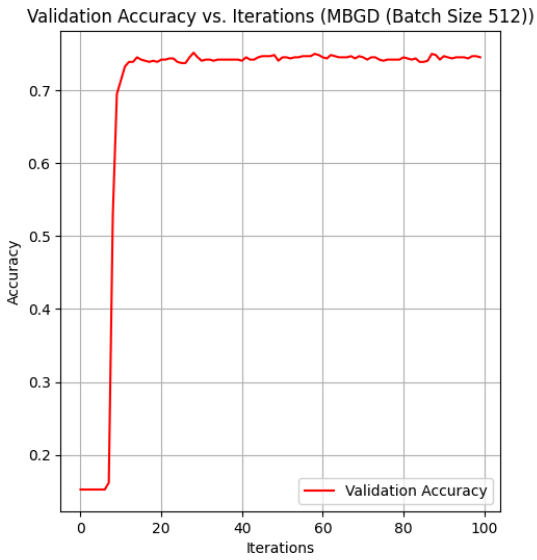

Training Accuracy vs. Iterations (MBGD (Batch Size 256))
Training Accuracy
Accuracy
Iterations
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

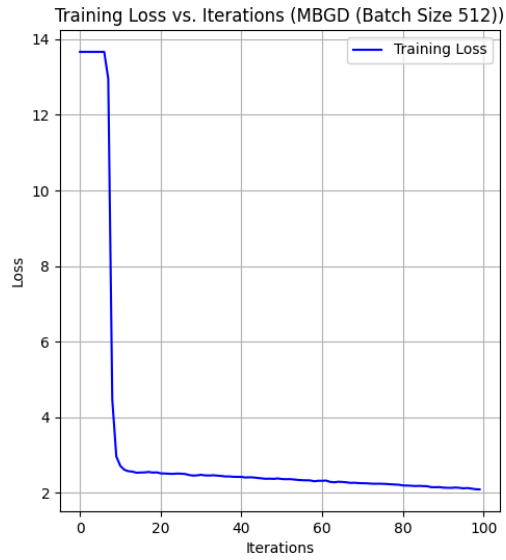

Validation Accuracy vs. Iterations (MBGD (Batch Size 256))
Validation Accuracy
Accuracy
Iterations
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
20
40
60
80
100

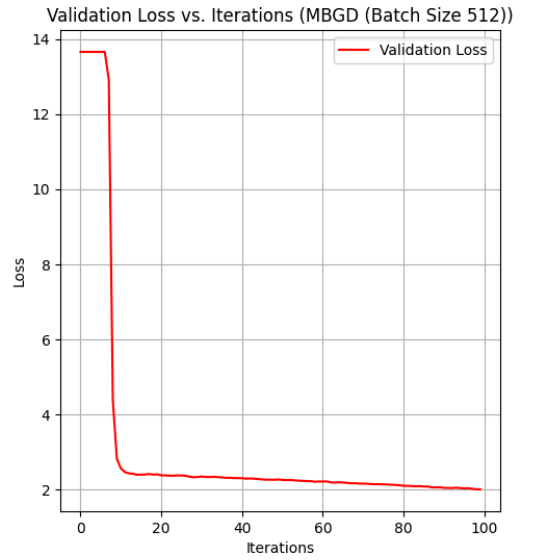

Training Loss vs. Iterations (MBGD (Batch Size 256))
Training Loss
Loss
Iterations
2
4
6
8
10
12
14
0
20
40
60
80
100


Validation Loss vs. Iterations (MBGD (Batch Size 256))
Validation Loss
Loss
Iterations
2
4
6
8
10
12
14
0
20
40
60
80
100









Learning Rate = 0.0001
Iterations = 100

Stochastic Gradient Descent and Mini-Batch Gradient Descent with 8, 32, 64, 256 & 512

1. Stochastic Gradient Descent

- **Training and Validation Accuracy**

    - The plots show significant fluctuations in both training and validation accuracy, ranging between **0.2** and **0.9** throughout the iterations. The accuracy fluctuates drastically with each iteration.
    - These large fluctuations are characteristic of SGD due to the use of a single data point for each update, which leads to unstable updates.
- **Training and Validation Loss**
    - The training and validation loss graphs exhibit large spikes and fluctuations, indicative of the noisy updates during optimization.
    - The loss does not stabilize and fluctuates between **1** and **6** across iterations, showing that the optimization is unstable.
- **Convergence Speed**: Faster due to per-sample updates, but noisy due to randomness.
- **Stability**: Very unstable with large fluctuations in both accuracy and loss, leading to less predictable learning.

2. Mini-Batch Gradient Descent - Batch Size 8

- **Training and Validation Accuracy**
    - MBGD with batch size 8 also experiences fluctuations, though not as pronounced as SGD. The accuracy moves between **0.6** and **0.8**, showing somewhat unstable learning.
- **Training and Validation Loss**
    - The training and validation loss decrease over time but exhibit periodic spikes, indicating some instability.
- **Convergence Speed**: Moderate speed, but still subject to fluctuations due to small batch size.
- **Stability**: Slightly better than SGD, though still subject to significant fluctuations.

3. Mini-Batch Gradient Descent - Batch Size 32

- **Training and Validation Accuracy (MBGD, Batch Size 32) (Second Plot Set)**:
    - The accuracy is more stable than with smaller batch sizes, fluctuating within a narrower range of **0.75** to **0.825**.
    - The fluctuations are still visible, but less pronounced than with a batch size of 8 or SGD.
- **Training and Validation Loss (MBGD, Batch Size 32) (Second Plot Set)**:
    - The loss decreases more steadily, with fewer spikes compared to smaller batch sizes.
- **Convergence Speed**: Balanced, moderate speed with fewer fluctuations.
- **Stability**: More stable compared to SGD and MBGD with a batch size of 8, though fluctuations still exist.

4. Mini-Batch Gradient Descent - Batch Size 64

- **Training and Validation Accuracy (MBGD, Batch Size 64) (Third Plot Set)**:

    - The accuracy stabilizes significantly, moving within a narrow range between **0.76** and **0.84**. Fewer fluctuations are seen compared to smaller batch sizes.
- **Training and Validation Loss (MBGD, Batch Size 64) (Third Plot Set)**:
    - The loss graph also stabilizes significantly with fewer spikes, indicating more predictable learning behavior.
- **Convergence Speed**: Balanced, moderate speed with further reduction in fluctuations.
- **Stability**: Improved stability, fewer fluctuations, and more predictable convergence.

5. Mini-Batch Gradient Descent - Batch Size 256

- **Training and Validation Accuracy (MBGD, Batch Size 256) (Fourth Plot Set)**:
    - The accuracy stabilizes early and remains close to **0.7**, showing minimal fluctuation.
- **Training and Validation Loss (MBGD, Batch Size 256) (Fourth Plot Set)**:
    - Both loss curves stabilize early and remain low, indicating that the model converges more quickly compared to smaller batch sizes.
- **Convergence Speed**: Faster convergence with minimal fluctuations.
- **Stability**: Highly stable with very minimal fluctuations.

6. Mini-Batch Gradient Descent - Batch Size 512

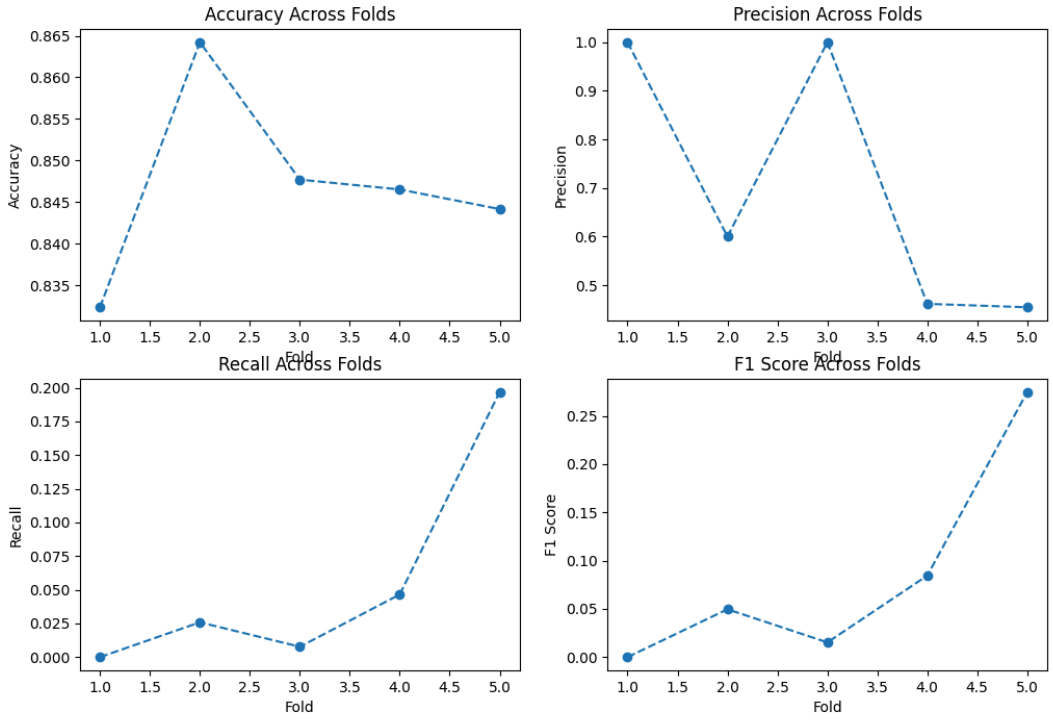
- **Training and Validation Accuracy (MBGD, Batch Size 512) (Fifth Plot Set)**:
    - The accuracy stabilizes extremely early and remains flat at around **0.7** throughout the iterations.
- **Training and Validation Loss (MBGD, Batch Size 512) (Fifth Plot Set)**:
    - Both loss curves are flat, showing early convergence and no fluctuations, indicating the highest stability among all batch sizes.
- **Convergence Speed**: Fastest convergence with minimal iterations needed.
- **Stability**: Highest stability, but the trade-off is reduced exploration of the optimization landscape.

Conclusion

1. **Convergence Speed**:
    - **SGD** converges quickly in the initial stages but is extremely noisy due to the randomness of single-sample updates.
    - **MBGD with larger batch sizes (256 and 512)** converges more steadily and rapidly, achieving stability earlier and with fewer fluctuations.
2. **Stability**:
    - **SGD** and **MBGD with small batch sizes (8 and 32)** exhibit significant fluctuations in both loss and accuracy due to the randomness introduced by small data subsets.
    - **MBGD with larger batch sizes (64, 256, 512)** offers much more stable convergence, with minimal fluctuations.
3. **Trade-offs**:

- **SGD and small batch MBGD**: Faster but noisier updates, which can help escape local minima but often results in instability.
- **Large batch MBGD**: Slower to update weights per iteration but offers much more predictable convergence, albeit at the risk of getting stuck in local minima.

e



Learning Rate = 0.0001
Iterations = 100

K-Fold Cross-Validation Results (k=5) for MBGD with Batch Size 8

1. Accuracy:

- **Mean = 0.8468 ± 0.0120**
  - The mean accuracy is quite high at **84.68%**, with a relatively low standard deviation of **0.0120**, indicating that the model performs consistently across the different folds.
  - **Stability:** The small variance in accuracy suggests that the model is stable and performs similarly on different subsets of data, reflecting robust generalization capabilities.

2. Precision:

- **Mean = 0.6851 ± 0.2595**
    - The precision shows a significant variance across folds, as indicated by the high standard deviation of **0.2595**. While the mean precision is **68.51%**, the high standard deviation implies that the model is inconsistent in its ability to correctly classify positive cases.
    - **Stability:** The large variance in precision suggests that the model's ability to avoid false positives is not stable across different data splits. This may indicate overfitting to certain folds or sensitivity to specific data points.

3. Recall:

- **Mean = 0.0610 ± 0.0526**
    - The mean recall is quite low at **6.10%**, with a moderate standard deviation of **0.0526**. This suggests that the model struggles to correctly identify positive cases across the different folds, and the variance reflects further instability in this regard.
    - **Stability:** The low recall across all folds indicates that the model is not effectively capturing positive cases and may be biased toward the negative class.

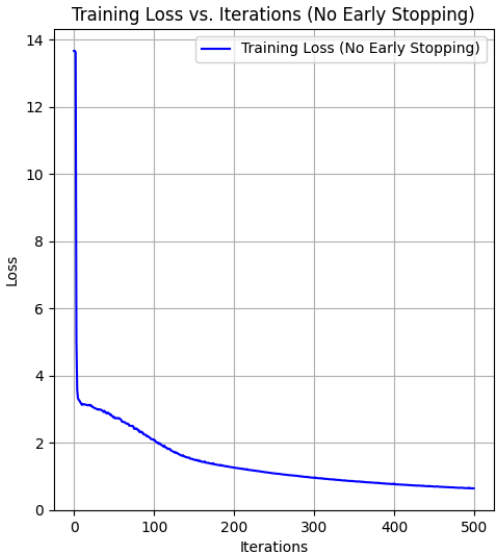
4. F1 Score:

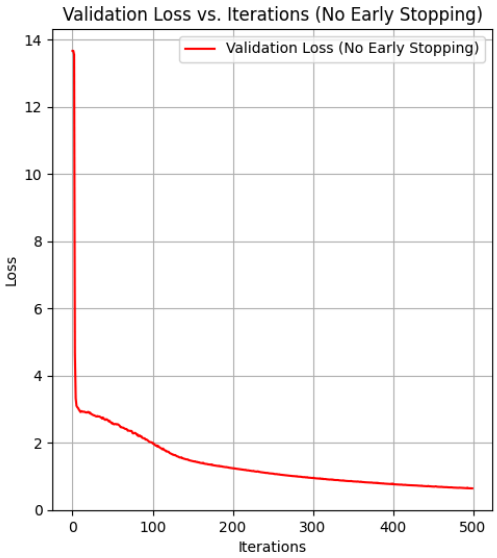
- **Mean = 0.0996 ± 0.0787**
    - The mean F1 score is low at **9.96%**, with a relatively high standard deviation of **0.0787**. This shows that the model has difficulty balancing precision and recall, and the performance fluctuates notably between different folds.
    - **Stability:** The high variance in the F1 score confirms that the model is unstable in its classification of positive cases, likely due to the low recall and fluctuating precision.

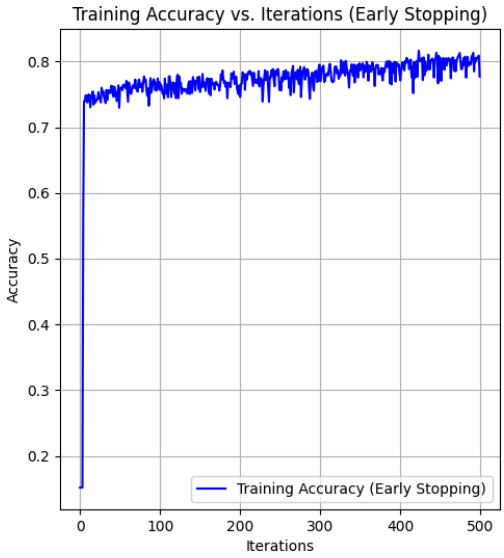
5. Stability and Variance:

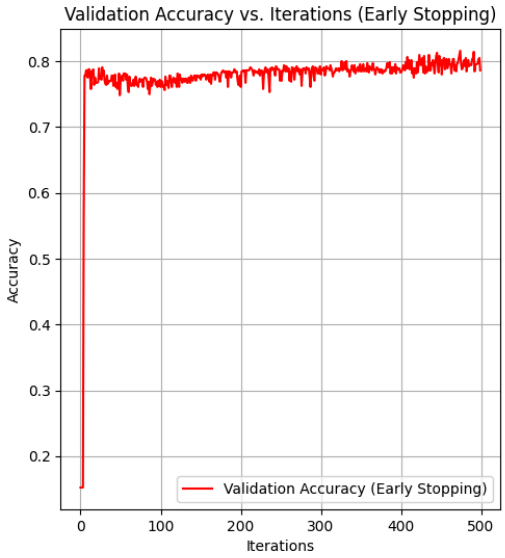
- **Accuracy Stability:** The model's accuracy is relatively stable across the folds, with a low variance. This suggests that the model generalizes well overall, though this may be due to its tendency to classify negative cases correctly.
- **Precision and Recall Variance:** The high variance in precision and the low recall indicate that the model's performance in identifying positive cases is inconsistent and unreliable. The model may be overfitting to certain folds and struggling to generalize its understanding of positive instances.
- **F1 Score Stability:** The low F1 score, coupled with high variance, reflects the model's struggle to achieve a good balance between precision and recall. This instability indicates that the model's performance fluctuates significantly depending on the data split.

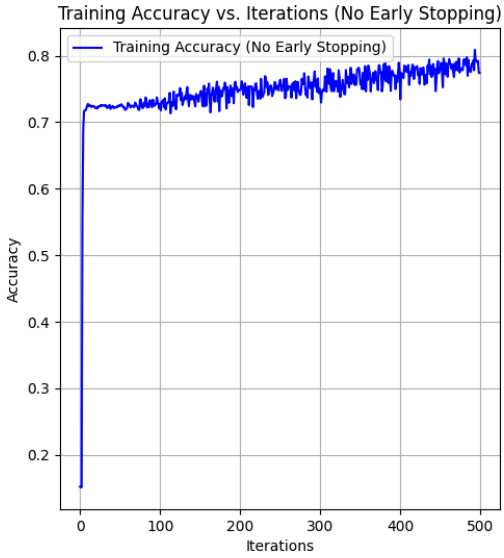
f.

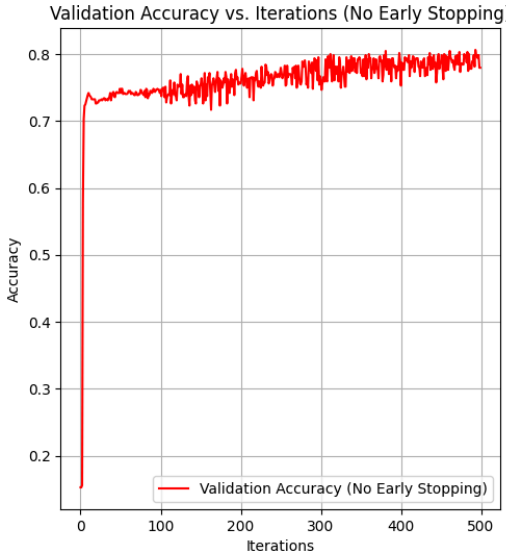

Training Loss vs. Iterations (No Early Stopping)
Training Loss (No Early Stopping)
Loss
Iterations
0
2
4
6
8
10
12
14
0
100
200
300
400
500

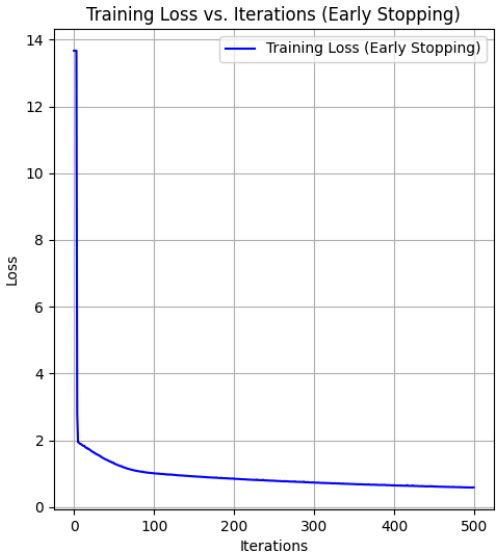

Validation Loss vs. Iterations (No Early Stopping)
Validation Loss (No Early Stopping)
Loss
Iterations
0
2
4
6
8
10
12
14
0
100
200
300
400
500

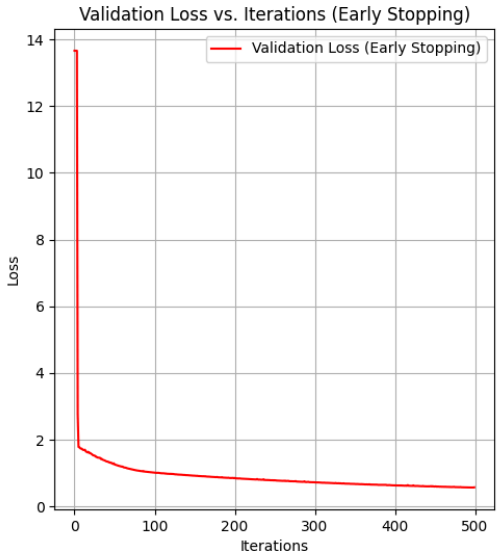

Training Accuracy vs. Iterations (Early Stopping)
Accuracy
Iterations
Training Accuracy (Early Stopping)


Validation Accuracy vs. Iterations (Early Stopping)
Accuracy
Iterations
Validation Accuracy (Early Stopping)


Training Accuracy vs. Iterations (No Early Stopping)
Accuracy
Iterations
Training Accuracy (No Early Stopping)


Validation Accuracy vs. Iterations (No Early Stopping)
Accuracy
Iterations
Validation Accuracy (No Early Stopping)





Learning Rate = 0.0001
Iterations = 500

Early Stopping in Gradient Descent

1. Training and Validation Accuracy with Early Stopping

- The accuracy plots with early stopping demonstrate that both training and validation accuracy steadily increase over iterations, but early stopping prevents overfitting by stopping the training once validation accuracy stops improving significantly.
- In the plots, we can see that both the training and validation accuracy stabilize around **0.78** to **0.80**. This indicates that the model is converging without significant overfitting due to early stopping. The fluctuations are minimal, which is a positive sign of stability.
- Early stopping halts the training before the model overfits, ensuring that the generalization error (performance on validation data) is minimized.

2. Training and Validation Accuracy without Early Stopping

- Without early stopping, the model continues training for a prolonged period, and both the training and validation accuracies improve slowly. However, as seen in the plots, there is more fluctuation in validation accuracy compared to the model with early stopping.
- The accuracy reaches a similar final value, but the model risks overfitting as it is not halted when the validation accuracy plateaus. Overfitting may occur after prolonged training, leading to a model that performs better on training data but struggles on unseen data.

3. Training and Validation Loss with Early Stopping

- The loss decreases rapidly in the beginning and then stabilizes at a low value (around **0.5**) as training progresses. The training loss continues to decrease, but the validation loss starts to stabilize, showing a good fit without overfitting.
- Early stopping ensures that once the validation loss stabilizes (indicating no further performance improvement on validation data), training halts to prevent the model from over-optimizing for training data.

4. Training and Validation Loss without Early Stopping

- The loss without early stopping continues to decrease throughout the training process, especially for the training loss. However, the validation loss begins to plateau much earlier, and continuing training beyond this point may lead to overfitting.
- The validation loss shows a more pronounced decrease in the initial stages but later becomes nearly constant, suggesting that further training does not improve the model's performance on unseen data.

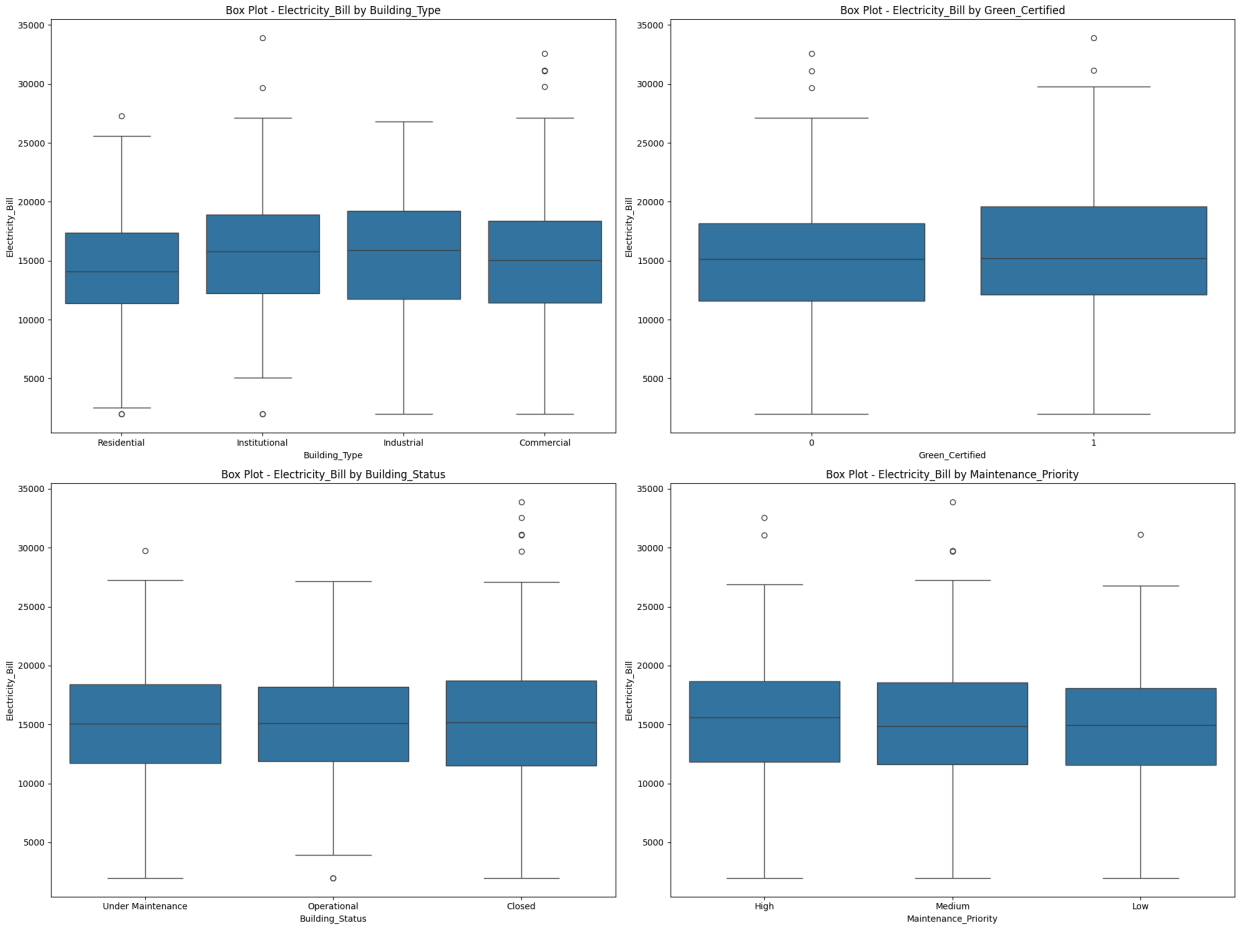
Early Stopping on Overfitting and Generalization

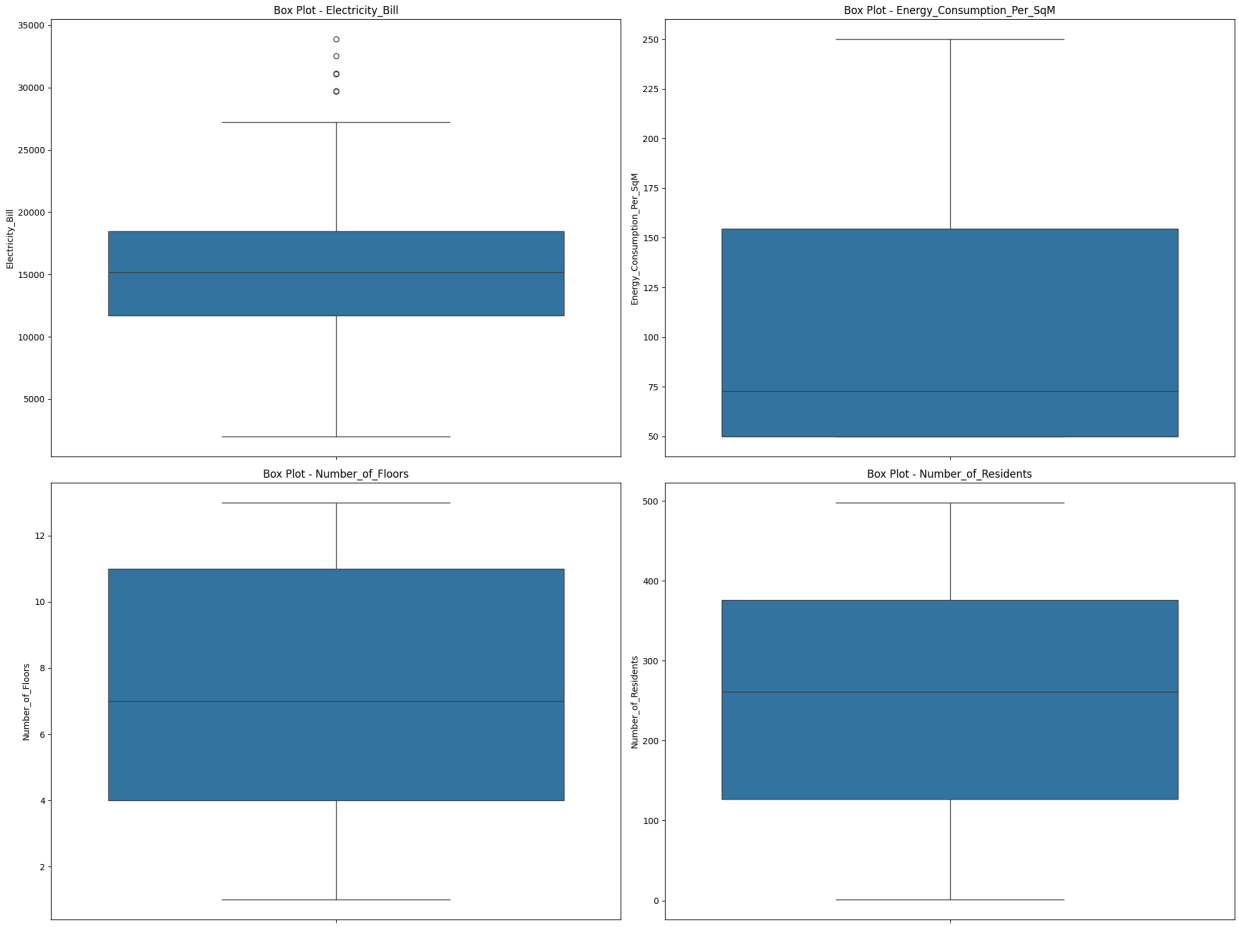
- **Effect on Overfitting:** Early stopping helps avoid overfitting by halting the training process when the validation loss plateaus, indicating that the model is not gaining any further benefit from additional iterations. This prevents the model from continuing to optimize for the training set at the expense of performance on unseen data.
- **Effect on Generalization:** Early stopping improves generalization by stopping the training at an optimal point where the validation accuracy and loss stabilize. By doing so, the model is better suited to perform on new, unseen data, which is a key measure of the model's robustness.

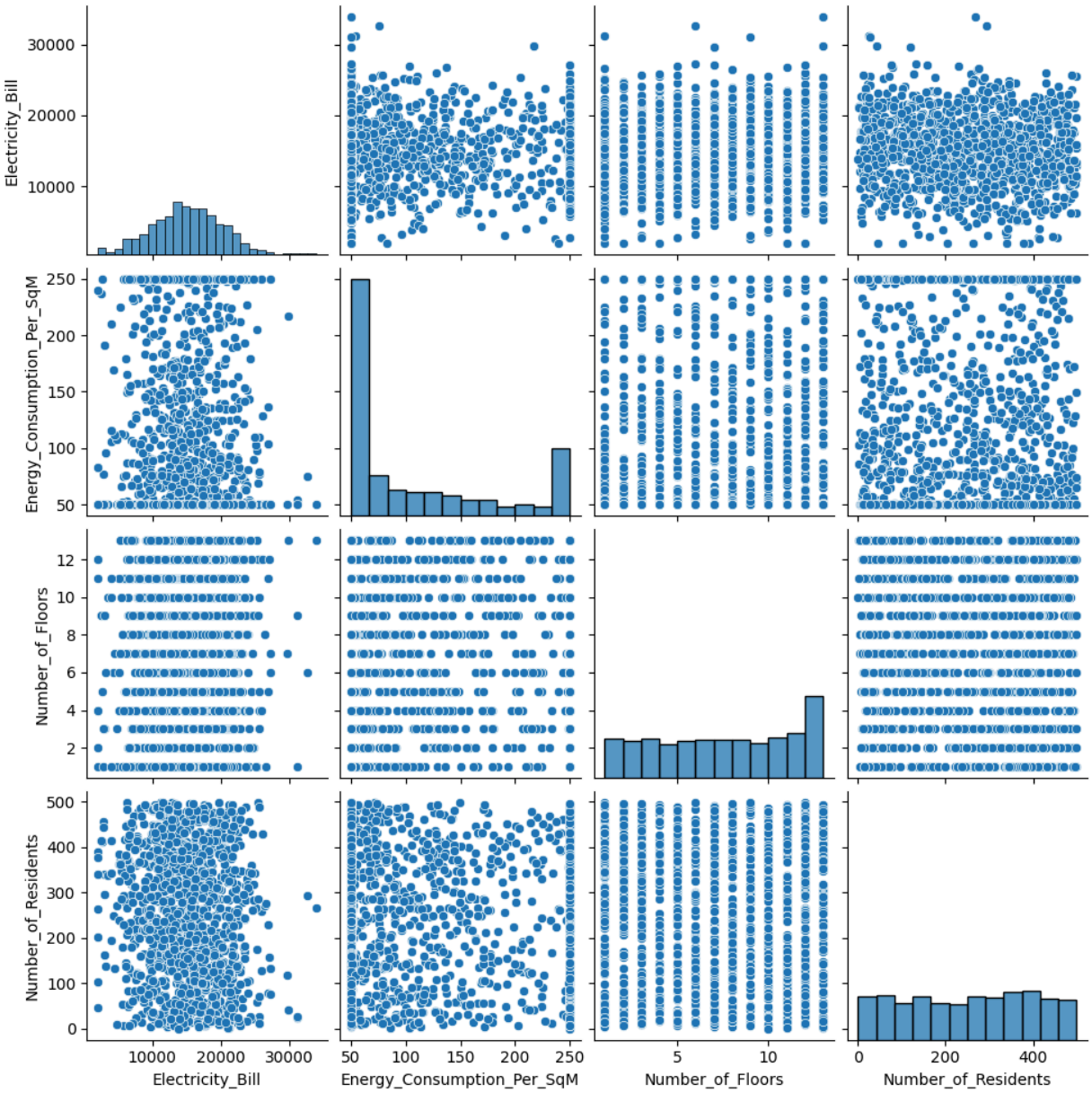
Conclusion

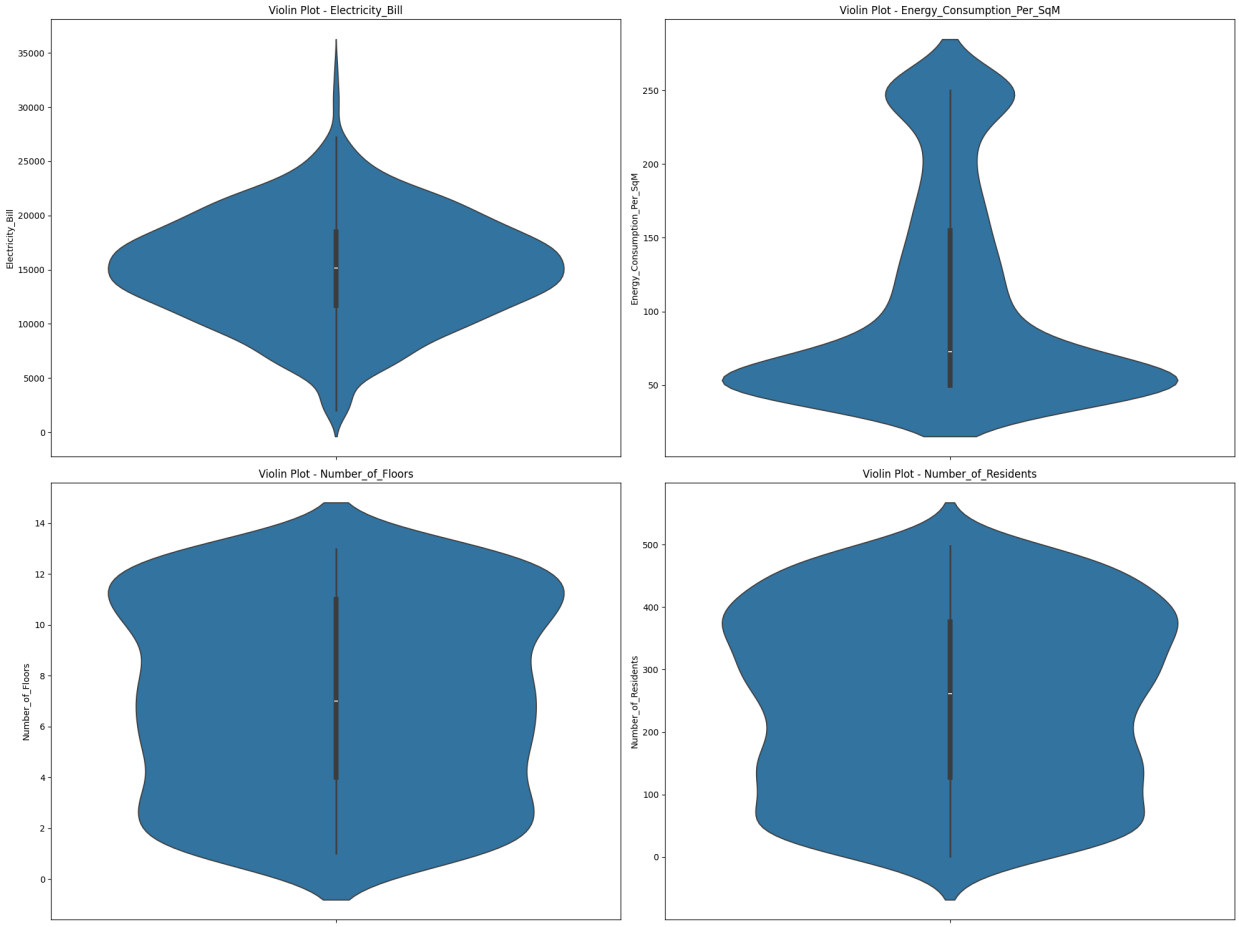
- **With Early Stopping**: The model is able to generalize better by preventing overfitting, stabilizing the validation loss, and keeping validation accuracy steady. Training is halted at an optimal point, balancing performance on both training and validation data.
- **Without Early Stopping**: The model runs the risk of overfitting as the training continues even after the validation accuracy stops improving, leading to potential degradation in performance on unseen data.

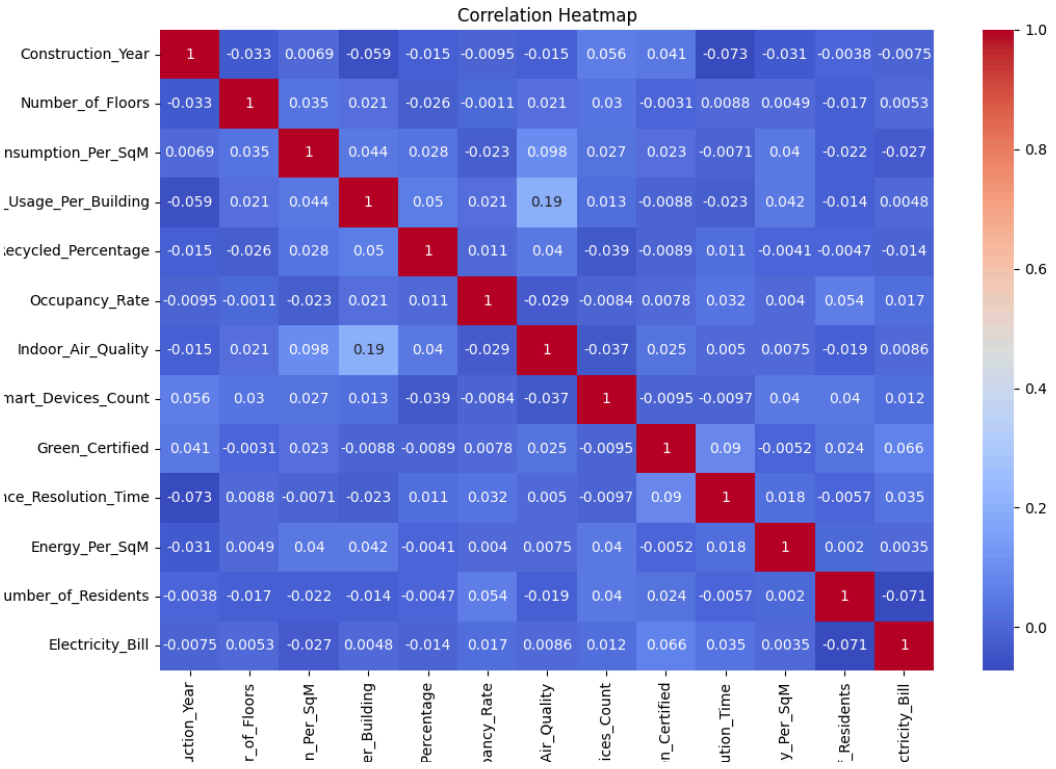
3. a.

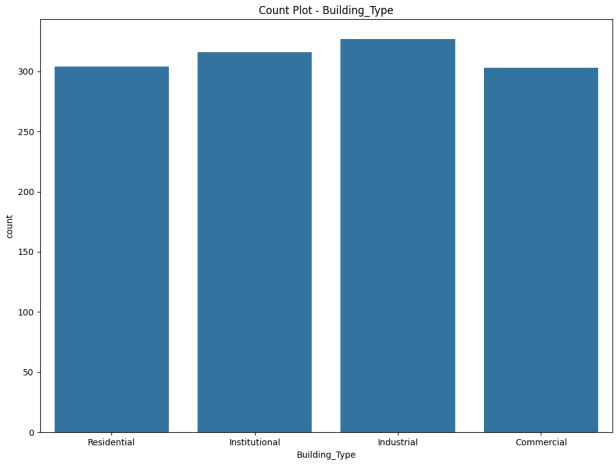

Box Plot - Electricity_Bill by Building_Type
Electricity_Bill
Building_Type
Residential
Institutional
Industrial
Commercial
Box Plot - Electricity_Bill by Green_Certified
Green_Certified
0
1
Box Plot - Electricity_Bill by Building_Status
Building_Status
Under Maintenance
Operational
Closed
Box Plot - Electricity_Bill by Maintenance_Priority
Maintenance_Priority
High
Medium
Low
35000
30000
25000
20000
15000
10000
5000

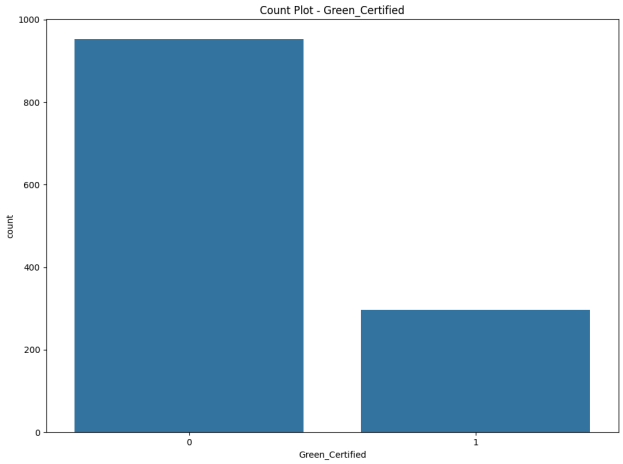

Box Plot - Electricity_Bill
Electricity_Bill
35000
30000
25000
20000
15000
10000
5000
Box Plot - Energy_Consumption_Per_SqM
Energy_Consumption_Per_SqM
250
225
200
175
150
125
100
75
50
Box Plot - Number_of_Floors
Number_of_Floors
12
10
8
6
4
2
Box Plot - Number_of_Residents
Number_of_Residents
500
400
300
200
100
0

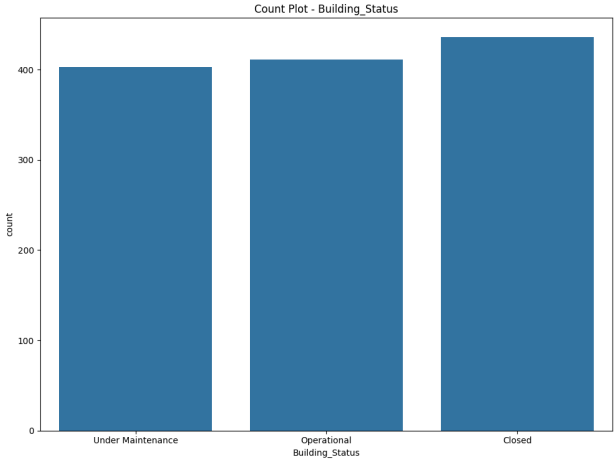

Electricity_Bill
Energy_Consumption_Per_SqM
Number_of_Floors
Number_of_Residents
30000
20000
10000
250
200
150
100
50
12
10
8
6
4
2
500
400
300
200
100
0
10000
20000
30000
50
100
150
200
250
5
10
0
200
400

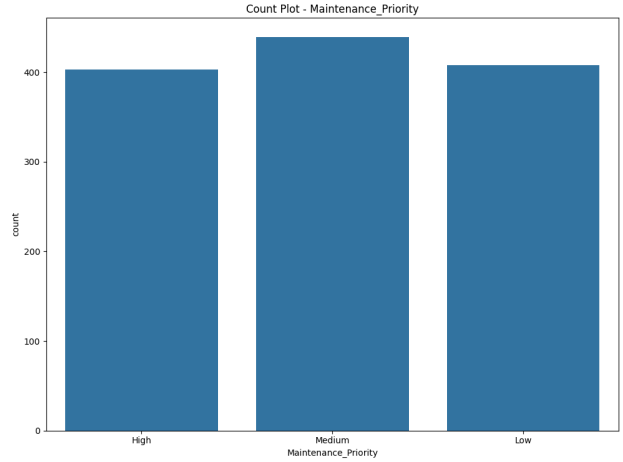

Violin Plot - Electricity_Bill
Electricity_Bill
35000
30000
25000
20000
15000
10000
5000
0
Violin Plot - Energy_Consumption_Per_SqM
Energy_Consumption_Per_SqM
250
200
150
100
50
Violin Plot - Number_of_Floors
Number_of_Floors
14
12
10
8
6
4
2
0
Violin Plot - Number_of_Residents
Number_of_Residents
500
400
300
200
100
0

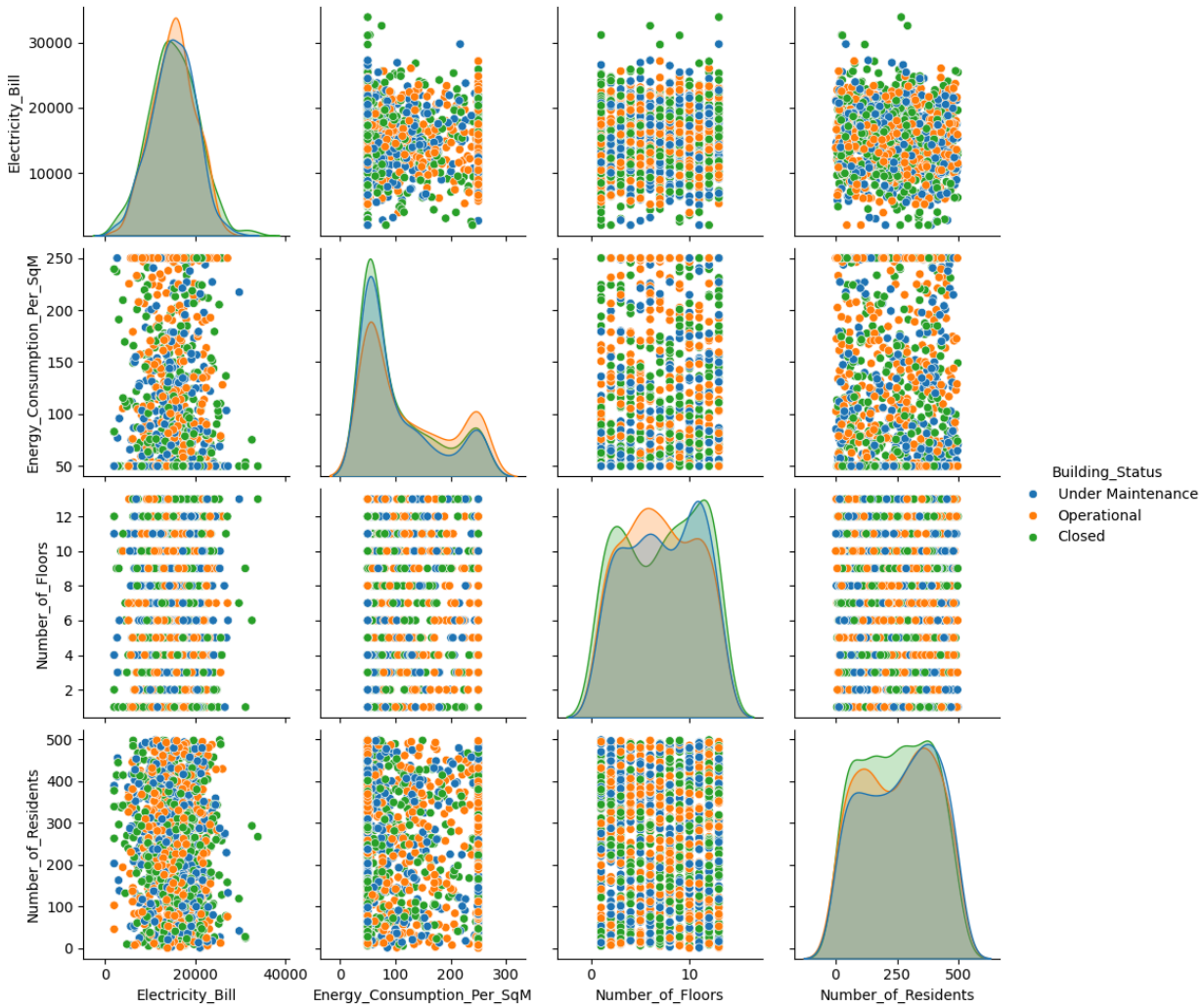

Correlation Heatmap
Construction_Year 1 -0.033 0.0069 -0.059 -0.015 -0.0095 -0.015 0.056 0.041 -0.073 -0.031 -0.0038 -0.0075
Number_of_Floors -0.033 1 0.035 0.021 -0.026 -0.0011 0.021 0.03 -0.0031 0.0088 0.0049 -0.017 0.0053
nsumption_Per_SqM 0.0069 0.035 1 0.044 0.028 -0.023 0.098 0.027 0.023 -0.0071 0.04 -0.022 -0.027
_Usage_Per_Building -0.059 0.021 0.044 1 0.05 0.021 0.19 0.013 -0.0088 -0.023 0.042 -0.014 0.0048
ecycled_Percentage -0.015 -0.026 0.028 0.05 1 0.011 0.04 -0.039 -0.0089 0.011 -0.0041 -0.0047 -0.014
Occupancy_Rate -0.0095 -0.0011 -0.023 0.021 0.011 1 -0.029 -0.0084 0.0078 0.032 0.004 0.054 0.017
Indoor_Air_Quality -0.015 0.021 0.098 0.19 0.04 -0.029 1 -0.037 0.025 0.005 0.0075 -0.019 0.0086
mart_Devices_Count 0.056 0.03 0.027 0.013 -0.039 -0.0084 -0.037 1 -0.0095 -0.0097 0.04 0.04 0.012
Green_Certified 0.041 -0.0031 0.023 -0.0088 -0.0089 0.0078 0.025 -0.0095 1 0.09 -0.0052 0.024 0.066
ce_Resolution_Time -0.073 0.0088 -0.0071 -0.023 0.011 0.032 0.005 -0.0097 0.09 1 0.018 -0.0057 0.035
Energy_Per_SqM -0.031 0.0049 0.04 0.042 -0.0041 0.004 0.0075 0.04 -0.0052 0.018 1 0.002 0.0035
umber_of_Residents -0.0038 -0.017 -0.022 -0.014 -0.0047 0.054 -0.019 0.04 0.024 -0.0057 0.002 1 -0.071
Electricity_Bill -0.0075 0.0053 -0.027 0.0048 -0.014 0.017 0.0086 0.012 0.066 0.035 0.0035 -0.071 1
uction_Year
r_of_Floors
n_Per_SqM
er_Building
Percentage
ancy_Rate
Air_Quality
ices_Count
n_Certified
ution_Time
y_Per_SqM
_Residents
ctricity_Bill
1.0
0.8
0.6
0.4
0.2
0.0

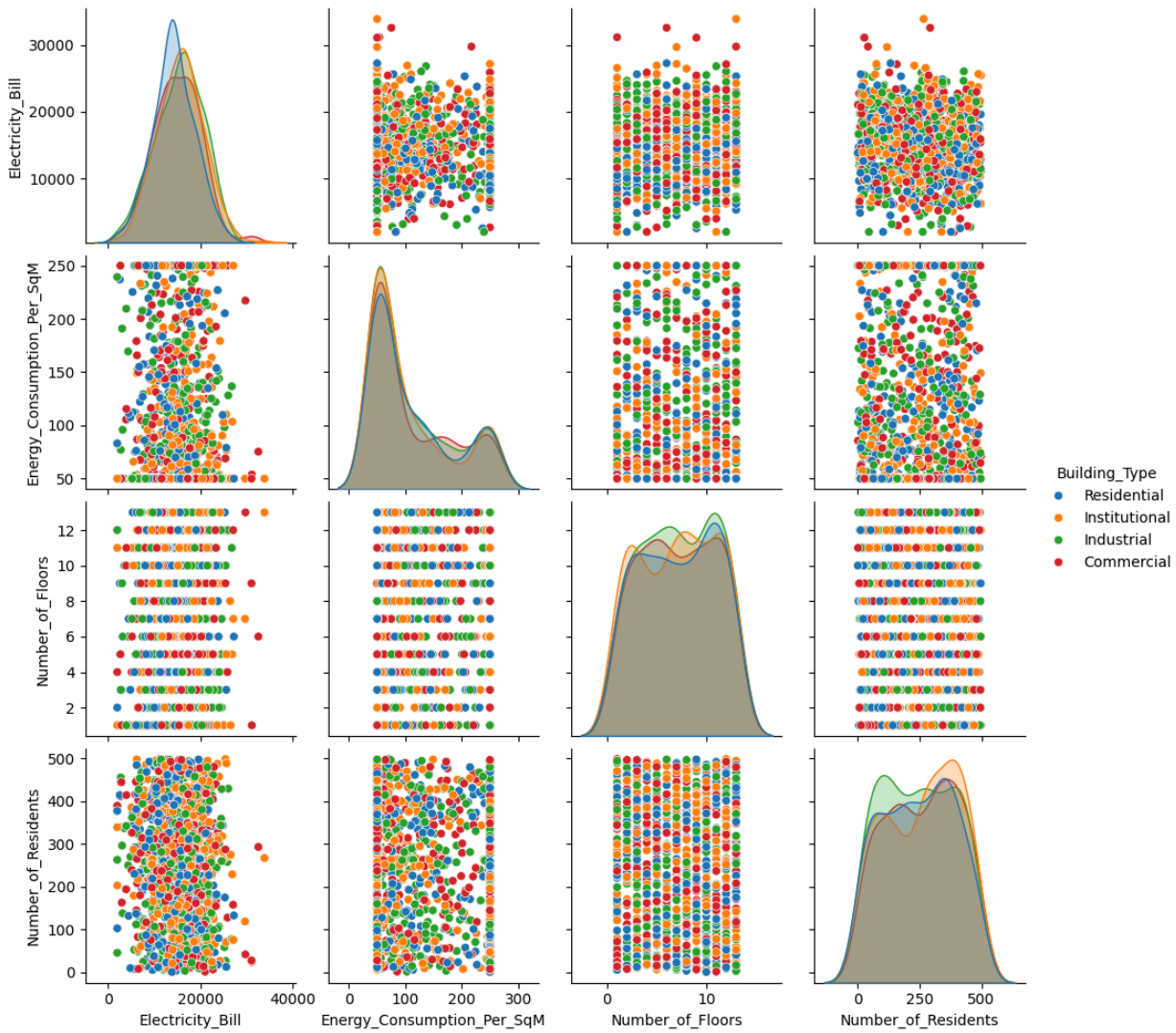

Count Plot - Building_Type
count
0
50
100
150
200
250
300
Residential
Institutional
Industrial
Commercial
Building_Type

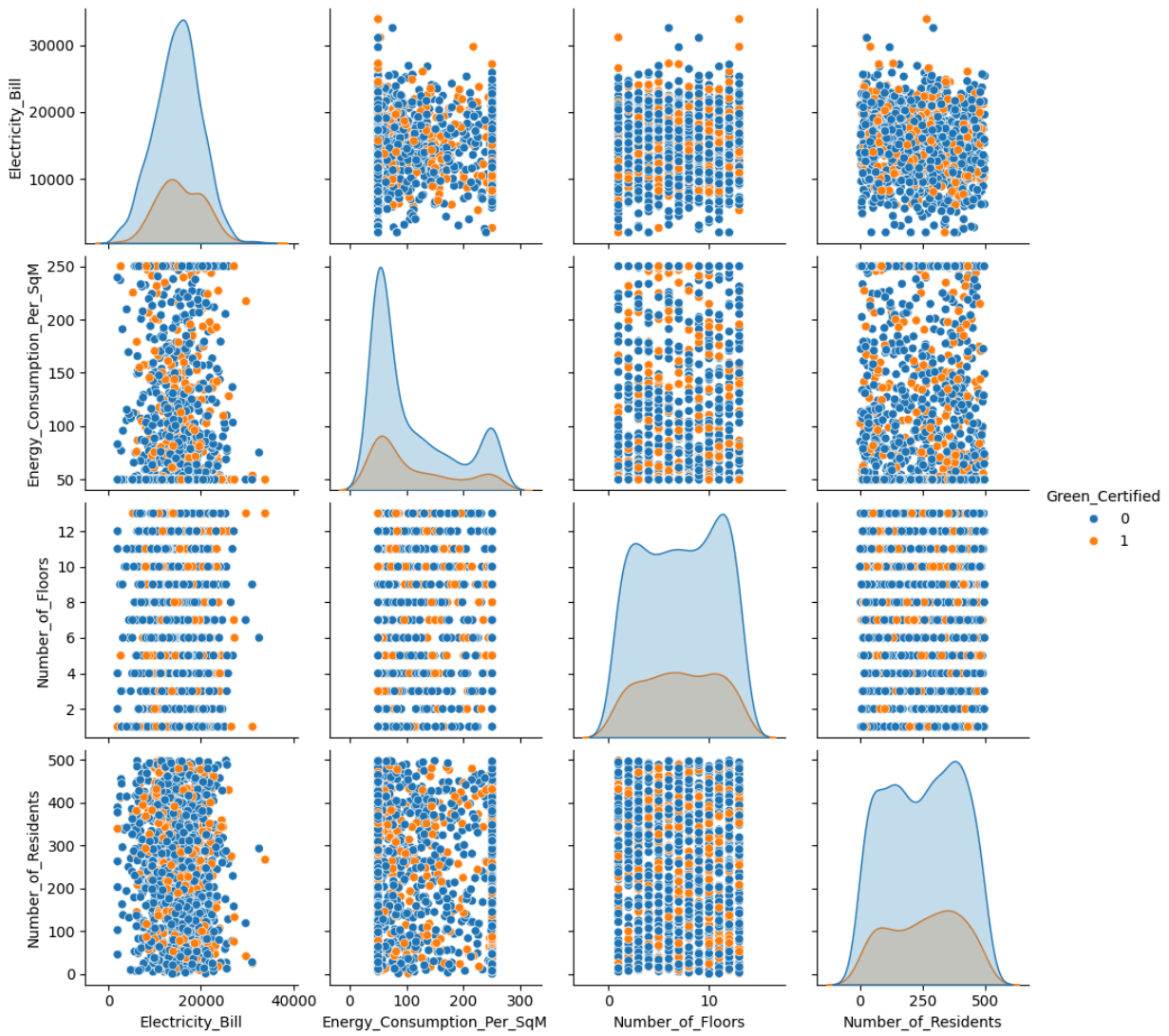

Count Plot - Green_Certified
count
0
200
400
600
800
1000
0
1
Green_Certified

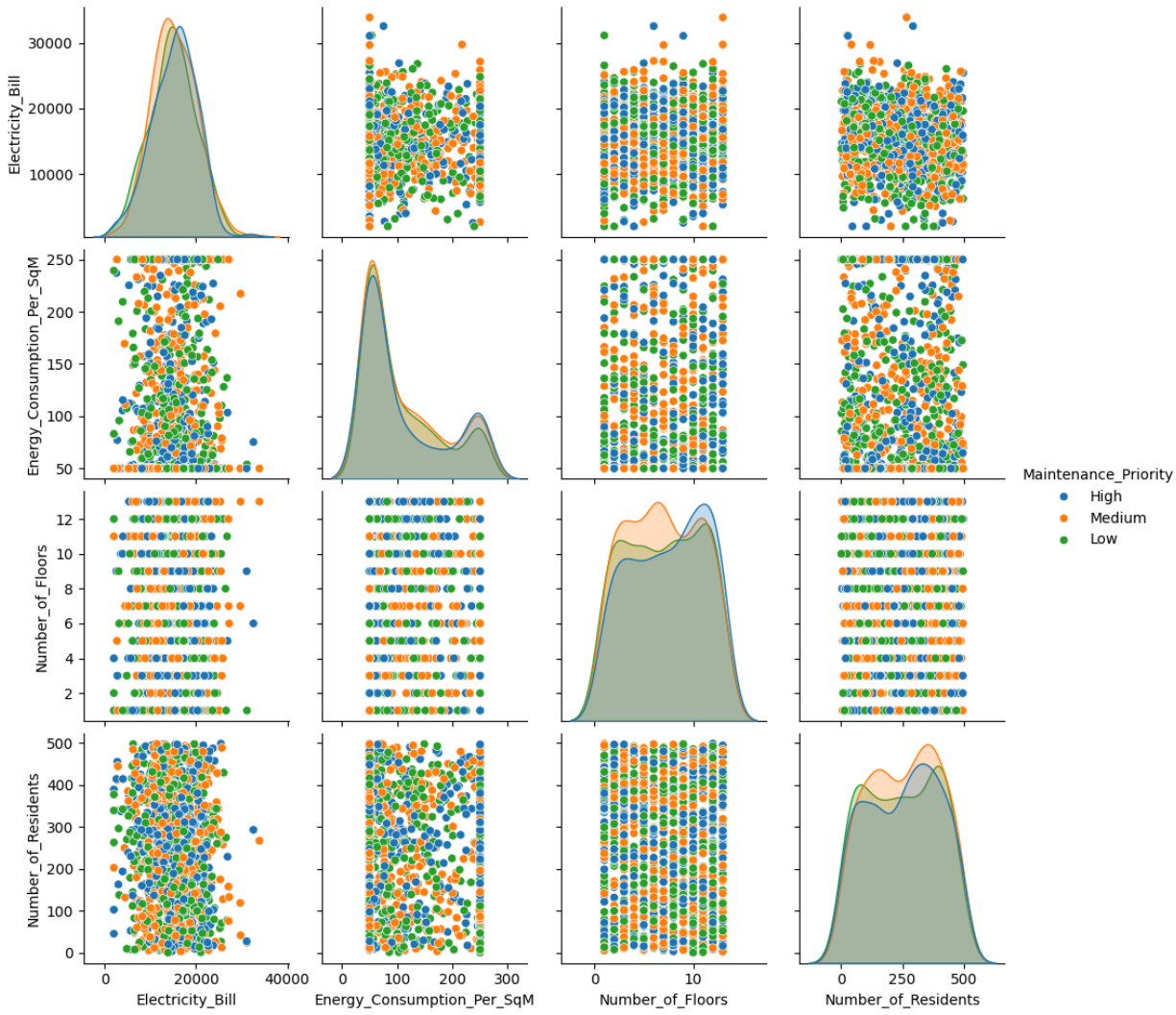

Count Plot - Building_Status
count
0
100
200
300
400
Under Maintenance
Operational
Closed
Building_Status

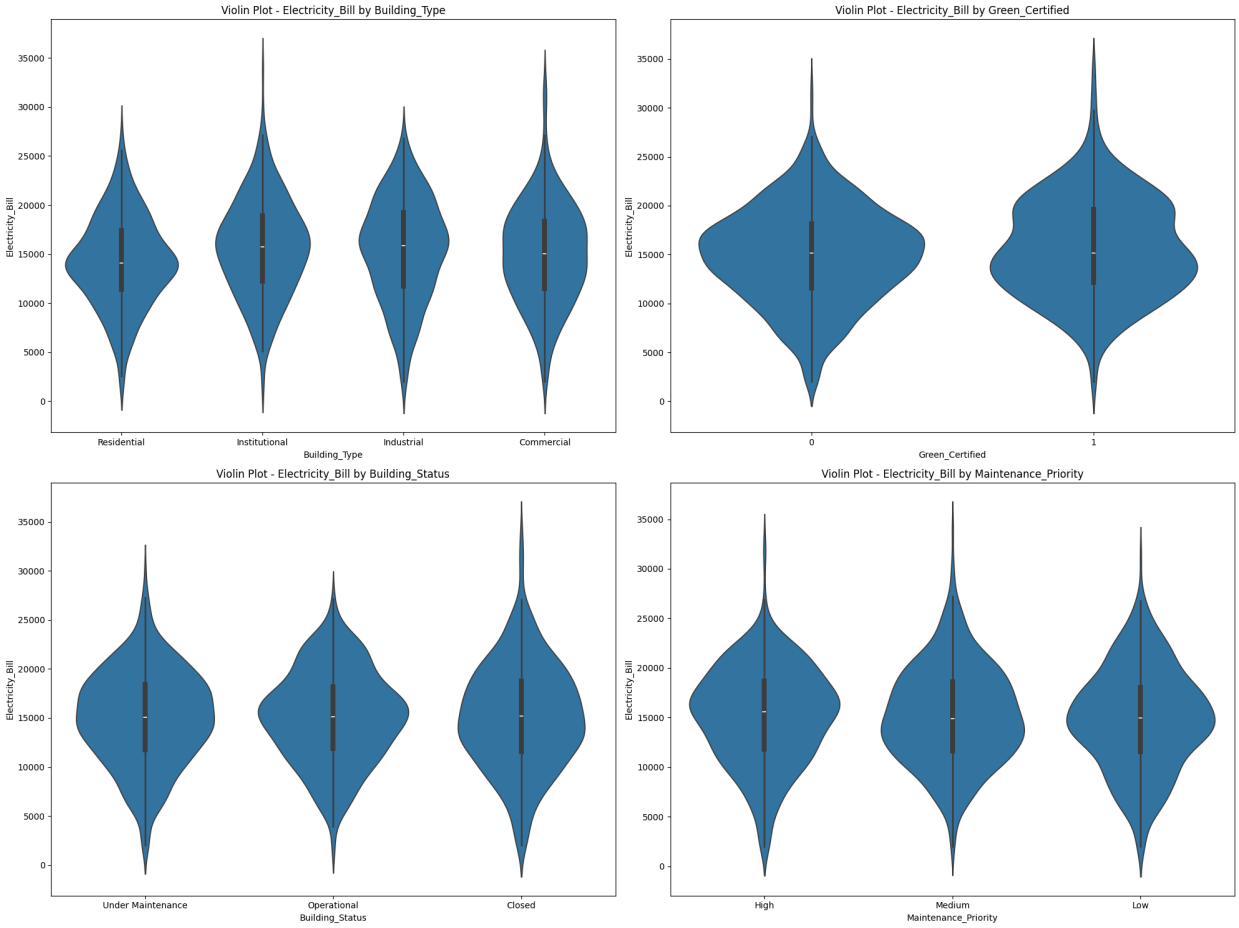

Count Plot - Maintenance_Priority
count
0
100
200
300
400
High
Medium
Low
Maintenance_Priority


Electricity_Bill
Energy_Consumption_Per_SqM
Number_of_Floors
Number_of_Residents
Building_Status
Under Maintenance
Operational
Closed


Electricity_Bill
Energy_Consumption_Per_SqM
Number_of_Floors
Number_of_Residents
Building_Type
Residential
Institutional
Industrial
Commercial


Electricity_Bill
Energy_Consumption_Per_SqM
Number_of_Floors
Number_of_Residents
Green_Certified
0
1


Electricity_Bill
Energy_Consumption_Per_SqM
Number_of_Floors
Number_of_Residents
Maintenance_Priority
High
Medium
Low



Building Type Distribution: The count plot of building types shows that the dataset is fairly balanced between Residential, Institutional, Industrial, and Commercial buildings. This suggests that the analysis is not skewed toward one particular type of building, providing a comprehensive view across building types.

Green Certification: From the count plot of the `Green_Certified` feature, it is evident that the majority of buildings in the dataset are not green certified (marked as 0), with only a small percentage being green certified. This might indicate limited implementation of energy-saving or environmentally friendly practices in the dataset.

Building Status: In the pair plots and violin plots, buildings that are operational tend to have lower electricity bills compared to buildings that are under maintenance or closed. This could suggest that operational buildings are more efficient in energy use.

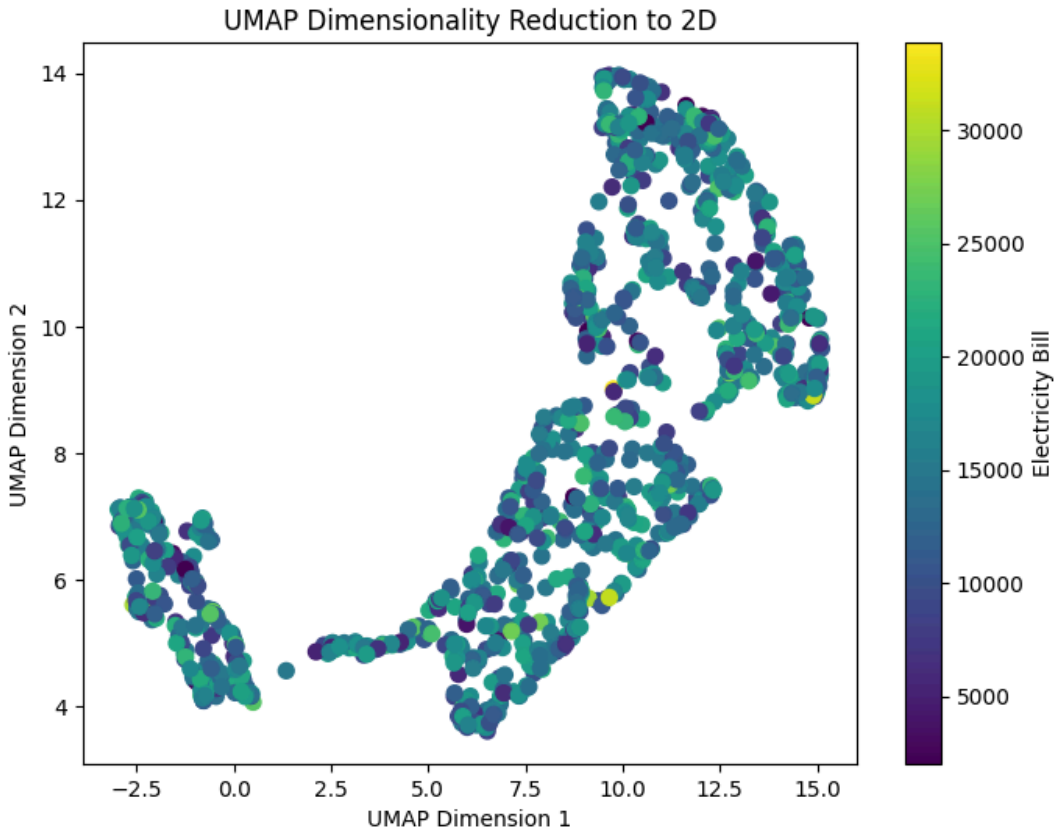
Maintenance Priority: Buildings with high maintenance priority generally have higher electricity bills, as seen in both the box and violin plots. This could be due to the fact that higher maintenance needs often correlate with older or less energy-efficient infrastructure.

Green Certification: The box and violin plots for green certification show that green-certified buildings tend to have slightly lower electricity bills, but the difference is not very substantial. However, this could indicate that green certification contributes somewhat to energy efficiency.

Weak Correlations: The correlation heatmap shows that there are generally weak correlations between most of the numerical features and the electricity bill. The strongest correlations are still weak (close to 0), suggesting that other factors (perhaps non-quantitative ones) might play a larger role in influencing electricity bills.

Electricity Consumption Distribution: The pair plot with `Building_Type` as the hue reveals that all building types exhibit a wide range of electricity bills, with commercial buildings slightly skewing towards higher electricity consumption. However, the distribution of electricity consumption per square meter is somewhat consistent across all types, indicating that other factors like building size or number of floors might influence the total bill more significantly than the building type alone.

b.



- After dimensionality reduction using UMAP, the data exhibits moderate clustering, with buildings that share similar characteristics—particularly those with lower electricity

bills—forming tighter groups. However, higher electricity bills are more dispersed across the projection, suggesting that high electricity consumption is spread across different building types and operational conditions, rather than concentrated in specific clusters.

- The separability between high and low electricity bills remains unclear, as there is significant overlap between regions. This indicates that multiple factors influence electricity usage, and these factors interact in complex ways. While UMAP preserves local structures effectively, the global separability of the data is limited, implying that further analysis is needed to fully understand the factors driving electricity consumption.

c.

```
PS C:\Users\vikra\OneDrive\Desktop\CSE343-ML\A1\Q3> python .\C.py

Number of Null Values in Each Column:

Building_Type                  0
Construction_Year              0
Number_of_Floors               0
Energy_Consumption_Per_SqM     0
Water_Usage_Per_Building       0
Waste_Recycled_Percentage      0
Occupancy_Rate                 0
Indoor_Air_Quality             0
Smart_Devices_Count            0
Green_Certified                0
Maintenance_Resolution_Time    0
Building_Status                0
Maintenance_Priority           0
Energy_Per_SqM                 0
Number_of_Residents            0
Electricity_Bill               0
dtype: int64

Train Metrics:
  MSE: 24475013.168475475
  RMSE: 4947.222773281538
  MAE: 4006.3284693293604
  R²: 0.0139225208446100 98
  Adjusted R²: -0.0011091480449538782

Test Metrics:
  MSE: 24278016.155742623
  RMSE: 4927.272689403604
  MAE: 3842.409312558516
  R²: 3.7344733075372893e-05
  Adjusted R²: -0.06406282547634 29
```

1. Handling Missing Values

As there were no missing values in the dataset, no further imputation or removal of records was needed, but still for safe side, we handled it by filling null values with mean.

2. Normalization of Numerical Features

Numerical features such as `Energy_Consumption_Per_SqM`, `Water_Usage_Per_Building`, `Waste_Recycled_Percentage`, and others were normalized to bring the features to a similar scale. This ensures that no particular feature dominates the model due to its magnitude.

3. Label Encoding for Categorical Features

Categorical columns such as `Building_Type` and `Building_Status` were label-encoded. Label encoding is used to convert these categorical features into numerical representations for the model to interpret effectively.

4. Linear Regression Model

A linear regression model was applied to the pre-processed data. The model was trained on the training set, and its performance was evaluated on both the training and testing sets.

Results

- **Train Metrics**
  - **MSE**: 24,475,013.17
  - **RMSE**: 4,947.22
  - **MAE**: 4,006.33
  - **R^2**: 0.0139
  - **Adjusted R^2**: -0.0011

- **Test Metrics**
  - **MSE**: 24,278,016.16
  - **RMSE**: 4,927.27
  - **MAE**: 3,842.41
  - **R^2**: 0.000037
  - **Adjusted R^2**: -0.0641

The performance of the linear regression model is relatively poor based on the results. The **R^2 score** and **Adjusted R^2 score** indicate that the model explains very little variance in both the training and testing data. Specifically:

- **R^2** values close to zero on both the training (0.0139) and testing (0.000037) sets suggest that the model does not fit the data well.
- The **Adjusted R^2** values are negative (-0.0011 and -0.0641), confirming that adding more predictors does not improve the model's performance.

The errors (MSE, RMSE, MAE) are relatively high, which further indicates that the model's predictions deviate significantly from the true values. The similar performance on both the training and test sets indicates that the model does not suffer from overfitting but might not be capturing the underlying relationships in the data.

This could be due to a lack of linear relationships between the features and the target variable, suggesting that more complex models or additional feature engineering may be necessary for improved performance.

d.

```
PS C:\Users\vikra\OneDrive\Desktop\CSE343-ML\A1\Q3> python .\D.py
Selected Features: Index(['Building_Type', 'Green_Certified', 'Building_Status'], dtype='object')
Train Metrics (with selected features):
  MSE: 24673540.31152836
  RMSE: 4967.246753638112
  MAE: 4006.784035347106
  R²: 0.005924030979948536
  Adjusted R²: 0.0029298262539845243

Test Metrics (with selected features):
  MSE: 24181190.647202764
  RMSE: 4917.437406536332
  MAE: 3825.6515746669897
  R²: 0.004025392685427787
  Adjusted R²: -0.008120639111091288
```

To improve the performance of the linear regression model, **Recursive Feature Elimination** was performed on the original dataset to select the 3 most important features. RFE is a feature selection method that recursively removes the least important features based on their contribution to the model's predictive power. The following features were selected:

- **Building_Type**
- **Green_Certified**
- **Building_Status**

After selecting these features, the linear regression model was re-trained, and its performance was evaluated on both the training and testing sets.

Results:

- **Train Metrics**
  - **MSE**: 24,673,540.31
  - **RMSE**: 4,967.25
  - **MAE**: 4,006.78
  - **R^2**: 0.0059

    - **Adjusted R^2**: 0.0029
- **Test Metrics**
    - **MSE**: 24,181,190.65
    - **RMSE**: 4,917.44
    - **MAE**: 3,825.65
    - **R^2**: 0.0040
    - **Adjusted R^2**: -0.0081

Comparison with Original Model:

Metric	Original Model (Train)	RFE Model (Train)	Original Model (Test)	RFE Model (Test)
MSE	24,475,013.17	24,673,540.31	24,278,016.16	24,181,190.65
RMSE	4,947.22	4,967.25	4,927.27	4,917.44
MAE	4,006.33	4,006.78	3,842.41	3,825.65
R^2	0.0139	0.0059	0.000037	0.0040
Adjusted R^2	-0.0011	0.0029	-0.0641	-0.0081

1. **Model Performance:**
    - The **R^2** and **Adjusted R^2** scores for the model using selected features are slightly lower for the training set (0.0059 vs. 0.0139) compared to the original model but show a marginal improvement on the test set (0.0040 vs. 0.000037).
    - The **MSE**, **RMSE**, and **MAE** values for both the training and test sets are relatively close between the original model and the model using selected features. However, there is a small reduction in the errors for the test set with the RFE model, particularly in **MAE** (3,825.65 vs. 3,842.41).
2. **Improvement in Generalization:**
    - The RFE model shows a slight improvement in the test set performance, as seen by a small increase in the R^2 score (from 0.000037 to 0.0040). However, this

improvement is minimal and suggests that the linear regression model may not be well-suited for this dataset, even with feature selection.

3. **Conclusion:**
    - While selecting the most important features helped reduce some of the errors, the overall improvement is not significant. This indicates that further experimentation with more advanced models or feature engineering might be necessary to achieve better predictive performance.

e.

```
PS C:\Users\vikra\OneDrive\Desktop\CSE343-ML\A1\Q3> python .\E.py
Train Metrics (Ridge Regression with One-Hot Encoding):
  MSE: 24188936.71529504
  RMSE: 4918.224955743183
  MAE: 3976.694459809908
  R²: 0.025448298005834 37
  Adjusted R²: 0.01059232693885015

Test Metrics (Ridge Regression with One-Hot Encoding):
  MSE: 24129257.316123597
  RMSE: 4912.154040349671
  MAE: 3797.611567420423
  R²: 0.006164422139467329
  Adjusted R²: -0.05754298669774638
```

One-Hot Encoding was applied to the categorical features of the original dataset, and **Ridge Regression** was performed on the preprocessed data. Ridge Regression introduces a penalty to the linear regression model to prevent overfitting by shrinking the coefficients of less important features.

It is used to transform categorical features into binary vectors. This approach allows the model to treat each category as a separate feature, which can improve performance when working with categorical data.

Results:

- **Train Metrics**
    - **MSE**: 24,188,936.72
    - **RMSE**: 4,918.22
    - **MAE**: 3,976.69
    - **R^2**: 0.0254
    - **Adjusted R^2**: 0.0106

- **Test Metrics**
    - **MSE**: 24,129,257.32

- **RMSE**: 4,912.15
- **MAE**: 3,797.61
- **R^2**: 0.0062
- **Adjusted R^2**: -0.0575

Comparison with Original Linear Regression Model

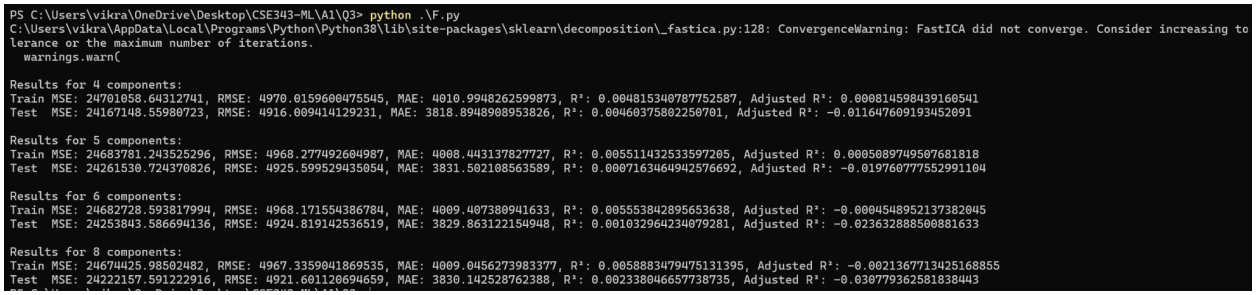
Metric	Original Model (Train)	Ridge Regression (Train)	Original Model (Test)	Ridge Regression (Test)
MSE	24,475,013.17	24,188,936.72	24,278,016.16	24,129,257.32
RMSE	4,947.22	4,918.22	4,927.27	4,912.15
MAE	4,006.33	3,976.69	3,842.41	3,797.61
R^2	0.0139	0.0254	0.000037	0.0062
Adjusted R^2	-0.0011	0.0106	-0.0641	-0.0575

Analysis

1. **Model Performance**
   - **R^2** and **Adjusted R^2** scores show a slight improvement with Ridge Regression on both the training and test sets compared to the original Linear Regression model.
   - The **MSE**, **RMSE**, and **MAE** values have decreased, especially in the test set, indicating that the Ridge Regression model is performing slightly better in reducing prediction errors.
2. **Effect of Regularization**
   - Ridge Regression adds a regularization term, which helps to penalize large coefficients and prevent overfitting. As a result, the **R^2** value improved from 0.000037 (original model) to 0.0062 on the test set. This suggests that Ridge Regression provides a better fit to the data, even though the improvement is marginal.

3. **Improvement in Generalization**
    - Although the improvements are relatively small, the Ridge Regression model generalizes slightly better to the test set. The errors (MSE, RMSE, MAE) are consistently lower compared to the original Linear Regression model, indicating that regularization helps the model perform better on unseen data.
4. **Conclusion**
    - The introduction of regularization through Ridge Regression combined with One-Hot Encoding improved the overall performance of the model slightly. The R^2 score is higher, and the error metrics are lower than the original Linear Regression model, particularly on the test data.

f.



Independent Component Analysis was applied to the one-hot encoded dataset to reduce dimensionality by transforming the data into independent components. ICA was performed with 4, 5, 6, and 8 components, and the linear regression model was trained and evaluated on the transformed data.

The results of ICA with different component numbers are compared based on the below.

Results

1. ICA with 4 Components

- **Train Metrics**
    - **MSE**: 24,701,058.64
    - **RMSE**: 4,970.02
    - **MAE**: 4,011.00
    - **R^2**: 0.0048
    - **Adjusted R^2**: 0.0008
- **Test Metrics**
    - **MSE**: 24,167,148.56
    - **RMSE**: 4,916.01
    - **MAE**: 3,818.89
    - **R^2**: 0.0046
    - **Adjusted R^2**: -0.0116

2. ICA with 5 Components

- **Train Metrics**
  - **MSE**: 24,683,781.24
  - **RMSE**: 4,968.28
  - **MAE**: 4,008.44
  - **R^2**: 0.0055
  - **Adjusted R^2**: 0.0005
- **Test Metrics**
  - **MSE**: 24,261,530.72
  - **RMSE**: 4,925.60
  - **MAE**: 3,831.50
  - **R^2**: 0.0007
  - **Adjusted R^2**: -0.0198

3. ICA with 6 Components

- **Train Metrics**
  - **MSE**: 24,682,728.59
  - **RMSE**: 4,968.17
  - **MAE**: 4,009.41
  - **R^2**: 0.0056
  - **Adjusted R^2**: -0.0005
- **Test Metrics**
  - **MSE**: 24,253,843.59
  - **RMSE**: 4,924.82
  - **MAE**: 3,829.86
  - **R^2**: 0.0010
  - **Adjusted R^2**: -0.0236

4. ICA with 8 Components

- **Train Metrics**
  - **MSE**: 24,674,425.99
  - **RMSE**: 4,967.34
  - **MAE**: 4,009.05
  - **R^2**: 0.0059
  - **Adjusted R^2**: -0.0021
- **Test Metrics**
  - **MSE**: 24,222,157.59
  - **RMSE**: 4,921.60
  - **MAE**: 3,830.14
  - **R^2**: 0.0023
  - **Adjusted R^2**: -0.0308

Comparison with Original Linear Regression Model (without ICA)

Metric	Original Model (Train)	ICA (4 Components)	ICA (5 Components)	ICA (6 Components)	ICA (8 Components)
MSE (Train)	24,475,013.17	24,701,058.64	24,683,781.24	24,682,728.59	24,674,425.99
RMSE (Train)	4,947.22	4,970.02	4,968.28	4,968.17	4,967.34
MAE (Train)	4,006.33	4,011.00	4,008.44	4,009.41	4,009.05
R^2 (Train)	0.0139	0.0048	0.0055	0.0056	0.0059
Adjusted R^2 (Train)	-0.0011	0.0008	0.0005	-0.0005	-0.0021
MSE (Test)	24,278,016.16	24,167,148.56	24,261,530.72	24,253,843.59	24,222,157.59
RMSE (Test)	4,927.27	4,916.01	4,925.60	4,924.82	4,921.60
MAE (Test)	3,842.41	3,818.89	3,831.50	3,829.86	3,830.14
R^2 (Test)	0.000037	0.0046	0.0007	0.0010	0.0023

Adjusted R^2 (Test)	-0.0641	-0.0116	-0.0198	-0.0236	-0.0308

Analysis

1. **Model Performance Across Components**
    - The **R^2** and **Adjusted R^2** scores remain low across all ICA components. While there is a slight improvement in the **R^2** score on the test set (up to 0.0046 with 4 components), it remains close to zero.
    - The error metrics (MSE, RMSE, MAE) for both the training and test sets are fairly consistent across all component numbers, indicating that the number of ICA components does not significantly impact the model's performance.
2. **Comparison with the Original Model**
    - The performance of the model with ICA is comparable to the original model. The **MSE**, **RMSE**, and **MAE** values are slightly lower with ICA, especially in the test set.
    - However, the improvement in **R^2** and **Adjusted R^2** is minimal, indicating that while ICA helps reduce dimensionality, it does not substantially improve the model's ability to explain variance in the data.
3. **Conclusion**
    - Applying ICA with different numbers of components did not lead to significant improvements in model performance. The **R^2** values remain low, and the errors are similar to those obtained with the original model. ICA may not be the most effective dimensionality reduction technique for this dataset, and other methods such as PCA or more advanced models may yield better results.

g.

```
PS C:\Users\vikra\OneDrive\Desktop\CSE343-ML\A1\Q3> python .\G.py

Results for alpha=0.1:
Train MSE: 24205653.883724913, RMSE: 4919.9241745909 98, MAE: 3977.798308794708, R²: 0.02477477774977769, Adjusted R²: 0.005867349714570 48
Test  MSE: 24115595.006779883, RMSE: 4910.763179667686, MAE: 3801.957382350426, R²: 0.006727145182436556, Adjusted R²: -0.07532582978075353

Results for alpha=0.05:
Train MSE: 24194282.967886765, RMSE: 4918.768440157228, MAE: 3976.9375107452 76, R²: 0.025232902032713334, Adjusted R²: 0.00633435625579659 16
Test  MSE: 24116734.872463692, RMSE: 4910.879236192201, MAE: 3800.1035601194735, R²: 0.006680196407523597 6, Adjusted R²: -0.07537665693272455

Results for alpha=0.01:
Train MSE: 24189197.729799714, RMSE: 4918.251491109388, MAE: 3976.687823565576, R²: 0.025437781953278282, Adjusted R²: 0.006543208338086748
Test  MSE: 24125941.950186923, RMSE: 4911.816563165497, MAE: 3798.0972807964936, R²: 0.006300975394228692, Adjusted R²: -0.07578720489929158

Results for alpha=0.005:
Train MSE: 24188995.71334019, RMSE: 4918.230953639752, MAE: 3976.6888140013407, R²: 0.025445921024737972, Adjusted R²: 0.006551505207870623
Test  MSE: 24127937.156726867, RMSE: 4912.019661679589, MAE: 3797.7978382905944, R²: 0.006218796849791985 4, Adjusted R²: -0.07587617210609476

Results for alpha=0.001:
Train MSE: 24188928.14838331, RMSE: 4918.224084807778, MAE: 3976.696312315222, R²: 0.025448643159426965, Adjusted R²: 0.006554280118640343
Test  MSE: 24129713.75647062, RMSE: 4912.20050043467, MAE: 3797.548726314041, R²: 0.006145622279608354, Adjusted R²: -0.07595539153207609
```

ElasticNet regularization was applied to the linear model. ElasticNet combines **L1 (Lasso)** and **L2 (Ridge)** regularization to handle multicollinearity and feature selection simultaneously. The model was trained on the preprocessed dataset, and results were reported for different values of the mixing parameter (**alpha**).

Results

1. **Alpha = 0.1**
    - **Train Metrics**
        - **MSE**: 24,205,653.88
        - **RMSE**: 4,919.92
        - **MAE**: 3,977.80
        - **R^2**: 0.0248
        - **Adjusted R^2**: 0.0059
    - **Test Metrics**
        - **MSE**: 24,115,595.01
        - **RMSE**: 4,910.76
        - **MAE**: 3,801.96
        - **R^2**: 0.0067
        - **Adjusted R^2**: -0.0753
2. **Alpha = 0.05**
    - **Train Metrics**
        - **MSE**: 24,194,282.97
        - **RMSE**: 4,918.77
        - **MAE**: 3,976.94
        - **R^2**: 0.0252
        - **Adjusted R^2**: 0.0063
    - **Test Metrics**
        - **MSE**: 24,116,734.87
        - **RMSE**: 4,910.88
        - **MAE**: 3,800.10
        - **R^2**: 0.0067
        - **Adjusted R^2**: -0.0754
3. **Alpha = 0.01**
    - **Train Metrics**
        - **MSE**: 24,189,197.73
        - **RMSE**: 4,918.25
        - **MAE**: 3,976.69
        - **R^2**: 0.0254
        - **Adjusted R^2**: 0.0065
    - **Test Metrics**
        - **MSE**: 24,125,941.95
        - **RMSE**: 4,911.82
        - **MAE**: 3,798.10
        - **R^2**: 0.0063
        - **Adjusted R^2**: -0.0758
4. **Alpha = 0.005**
    - **Train Metrics**
        - **MSE**: 24,188,995.71

        - **RMSE**: 4,918.23
        - **MAE**: 3,976.69
        - **R^2**: 0.0254
        - **Adjusted R^2**: 0.0066
    - **Test Metrics**
        - **MSE**: 24,127,937.16
        - **RMSE**: 4,912.02
        - **MAE**: 3,797.80
        - **R^2**: 0.0062
        - **Adjusted R^2**: -0.0759
5. **Alpha = 0.001**
    - **Train Metrics**
        - **MSE**: 24,188,928.15
        - **RMSE**: 4,918.22
        - **MAE**: 3,976.70
        - **R^2**: 0.0254
        - **Adjusted R^2**: 0.0066
    - **Test Metrics**
        - **MSE**: 24,129,713.76
        - **RMSE**: 4,912.20
        - **MAE**: 3,797.55
        - **R^2**: 0.0061
        - **Adjusted R^2**: -0.0759

Comparison with Original Linear Regression Model

Metric	Original Model (Train)	Alpha = 0.1	Alpha = 0.05	Alpha = 0.01	Alpha = 0.005	Alpha = 0.001
MSE (Train)	24,475,013.17	24,205,653.88	24,194,282.97	24,189,197.73	24,188,995.71	24,188,928.15
RMSE (Train)	4,947.22	4,919.92	4,918.77	4,918.25	4,918.23	4,918.22
MAE (Train)	4,006.33	3,977.80	3,976.94	3,976.69	3,976.69	3,976.70

R^2 (Train)	0.0139	0.0248	0.0252	0.0254	0.0254	0.0254
Adjusted R^2 (Train)	-0.0011	0.0059	0.0063	0.0065	0.0066	0.0066
MSE (Test)	24,278,016.16	24,115,595.01	24,116,734.87	24,125,941.95	24,127,937.16	24,129,713.76
RMSE (Test)	4,927.27	4,910.76	4,910.88	4,911.82	4,912.02	4,912.20
MAE (Test)	3,842.41	3,801.96	3,800.10	3,798.10	3,797.80	3,797.55
R^2 (Test)	0.000037	0.0067	0.0067	0.0063	0.0062	0.0061
Adjusted R^2 (Test)	-0.0641	-0.0753	-0.0754	-0.0758	-0.0759	-0.0759

Analysis

1. **Model Performance**
    - **R^2** and **Adjusted R^2** scores show marginal improvement compared to the original Linear Regression model across all values of **alpha**. For instance, the **R^2** score on the test set increases slightly from 0.000037 (original model) to 0.0067 with **alpha = 0.1**.
    - The **MSE**, **RMSE**, and **MAE** are slightly reduced across all values of **alpha** when compared to the original model, with the best results seen with **alpha = 0.1**.
2. **Impact of ElasticNet Regularization**

- The regularization introduced by ElasticNet (a combination of L1 and L2) helps in reducing overfitting and improving generalization slightly, as evidenced by the lower error values and higher **R²** scores across different values of **alpha**.
- The impact of different **alpha** values is relatively small, with the performance remaining fairly consistent across the range of values tested. However, **alpha = 0.1** provides the best overall balance in terms of lower error values and improved **R²** scores.

3. **Conclusion**
   - ElasticNet regularization improves the performance of the model slightly compared to the original Linear Regression model. The model with **alpha = 0.1** performs best in terms of **R²**, **MSE**, and **MAE**.
   - Although the improvements are modest, the regularization helps in better generalization, reducing the errors on the test set.

h.

```
PS C:\Users\vikra\OneDrive\Desktop\CSE343-ML\A1\Q3> python .\H.py
Train Metrics (Gradient Boosting Regressor):
  MSE: 15548098.780395458
  RMSE: 3943.1077566299728
  MAE: 3155.777526146695
  R²: 0.37358031452342877
  Adjusted R²: 0.3614354430703116

Test Metrics (Gradient Boosting Regressor):
  MSE: 24721402.6033069
  RMSE: 4972.06220831024
  MAE: 3829.578708527433
  R²: -0.0182248512623282
  Adjusted R²: -0.10233907810573784
```

Gradient Boosting Regressor was applied to the preprocessed dataset. Gradient Boosting is an ensemble learning method that combines multiple weak learners (in this case, decision trees) to create a stronger predictive model. The model was trained, and the performance was evaluated using the same metrics as in the previous parts: **MSE**, **RMSE**, **R²**, **Adjusted R²**, and **MAE**.

Gradient Boosting Regressor Results

- **Train Metrics**
  - **MSE**: 15,548,098.78
  - **RMSE**: 3,943.11
  - **MAE**: 3,155.78
  - **R²**: 0.3736
  - **Adjusted R²**: 0.3614
- **Test Metrics**
  - **MSE**: 24,721,402.60
  - **RMSE**: 4,972.06

- **MAE**: 3,829.58
- **R^2**: -0.0182
- **Adjusted R^2**: -0.1023

Comparison with Original Linear Regression Model (part c)

Metric	Original Model (Train)	Original Model (Test)	Gradient Boosting (Train)	Gradient Boosting (Test)
MSE	24,475,013.17	24,278,016.16	15,548,098.78	24,721,402.60
RMSE	4,947.22	4,927.27	3,943.11	4,972.06
MAE	4,006.33	3,842.41	3,155.78	3,829.58
R^2	0.0139	0.000037	0.3736	-0.0182
Adjusted R^2	-0.0011	-0.0641	0.3614	-0.1023

Comparison with ElasticNet Model (part g)

Metric	ElasticNet (Train, alpha=0.1)	ElasticNet (Test, alpha=0.1)	Gradient Boosting (Train)	Gradient Boosting (Test)
MSE	24,205,653.88	24,115,595.01	15,548,098.78	24,721,402.60
RMSE	4,919.92	4,910.76	3,943.11	4,972.06

MAE	3,977.80	3,801.96	3,155.78	3,829.58
R^2	0.0248	0.0067	0.3736	-0.0182
Adjusted R^2	0.0059	-0.0753	0.3614	-0.1023

Analysis:

1. **Model Performance**
    - The **Gradient Boosting Regressor** significantly outperforms the Linear Regression and ElasticNet models on the **training set**. The **MSE**, **RMSE**, and **MAE** are much lower, and the **R^2 score** is substantially higher at **0.3736**, indicating a better fit to the training data.
    - However, the model does not generalize well to the **test set**. The **R^2 score** on the test set is negative (-0.0182), which indicates that the model performs worse than a simple mean prediction model. This could be a sign of overfitting.
2. **Overfitting**
    - The large gap between the training and test performance suggests overfitting. Gradient Boosting tends to overfit when the model complexity is high or when the model is not properly regularized. In this case, the model performs very well on the training data but struggles on the test data.
3. **Comparison with ElasticNet and Linear Regression**
    - **Training Set:** The Gradient Boosting model performs much better on the training set compared to both ElasticNet and Linear Regression, with a significantly lower **MSE** and higher **R^2**.
    - **Test Set:** While ElasticNet shows a slight improvement over Linear Regression on the test set, the Gradient Boosting model performs worse on the test set, despite its strong performance on the training set.
4. **Conclusion**
    - The Gradient Boosting Regressor exhibits overfitting, achieving excellent results on the training set but failing to generalize well to the test set. Fine-tuning the model parameters, such as the learning rate, the number of boosting stages, or implementing early stopping, could help reduce overfitting and improve performance on the test set.
    - In comparison to ElasticNet and Linear Regression, Gradient Boosting offers more flexibility and potential for higher accuracy but requires careful tuning to prevent overfitting.