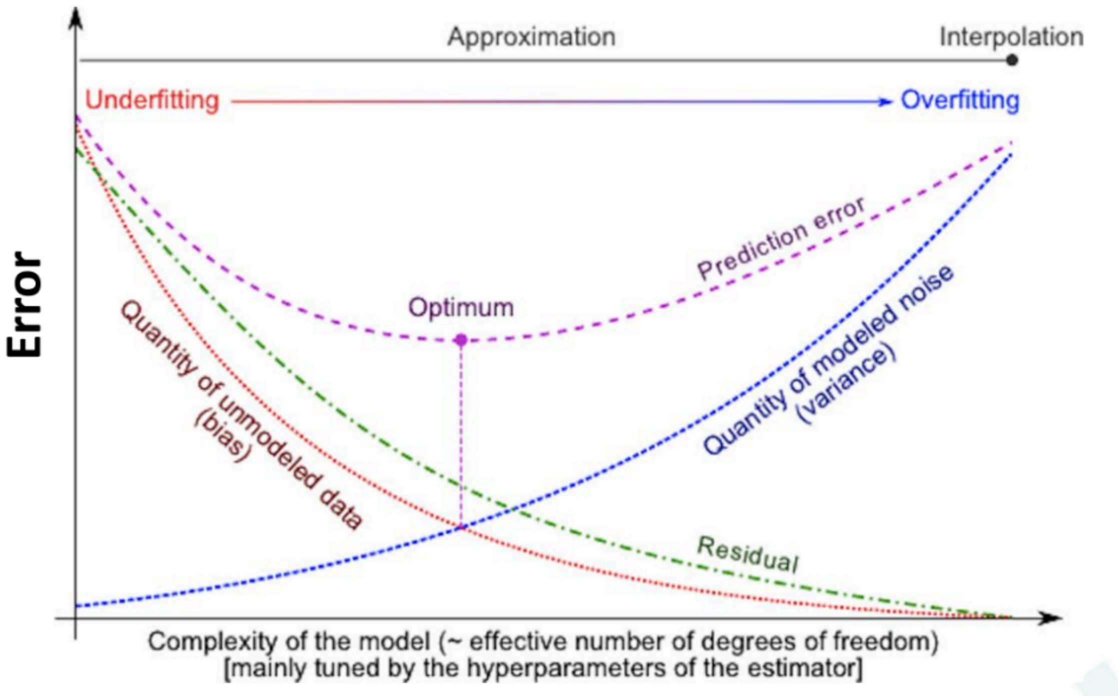
Assignment 1

Q1: As shown in the Bias-Variance Tradeoff Graph:



Low Complexity (High Bias, Low Variance): A simpler model, like linear regression with few features, tends to have high bias because it assumes a linear relationship between the input and output. This leads to **underfitting**: the model fails to capture the complexity of the data.
Increasing Complexity (Decreasing Bias, Increasing Variance): As you add more features or polynomial terms, the model becomes more flexible, reducing bias because it can better fit the training data. However, the model also becomes more sensitive to the noise in the training data, leading to increased variance and **overfitting**.
Very High Complexity (Low Bias, High Variance): In extreme cases, a highly complex model might capture almost every detail in the training set, even the noise, resulting in very low bias but extremely high variance. The model performs poorly on unseen data due to overfitting.

As you increase model complexity:

- **Bias** decreases.
- **Variance** increases. The goal is to find the right level of complexity that minimizes the test error by balancing bias and variance.

Q2:

True Positives (TP): Correctly identified spam emails = 200
False Negatives (FN): Spam emails that were classified as legitimate = 50
True Negatives (TN): Correctly identified legitimate emails = 730
False Positives (FP): Legitimate emails that were classified as spam = 20

1. Precision (spam detection):

   Precision = (TP) / (TP + FP) =200/(200 +20) = 0.909

2. Recall (spam):

   Recall = TP/ (TP+ FN) =200/ (200 + 50) = 0.8

3. Accuracy (overall classifications):

   Accuracy = (TP + TN)/(TP + TN + FP + FN) = (200 + 730) /(200 + 730 + 20 + 50) = 0.93

4. F1 Score = 2 * (Precision * Recall)/ (Precision + Recall)

   = 2(0.909 * 0.8)/ (0.909 + 0.8)
   = 0.85

Q3:

c).

x	y
3	15
6	30
10	55
15	85
18	100

$\Rightarrow$

$$X = \mathbb{R}^{n \times (d+1)}$$
$$\theta = R^{(d+1) \times 1}$$
$$Y = R^{n \times 1}$$

$$X = \begin{bmatrix} 3 & 1 \\ 6 & 1 \\ 10 & 1 \\ 15 & 1 \\ 18 & 1 \end{bmatrix}$$

$$Y = X\theta$$
$$\theta = (X^T X)^{-1} X^T Y$$

$$\theta = (X^T X)^{-1} X^T Y$$

$$= \begin{bmatrix} 3 & 6 & 10 & 15 & 18 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 3 & 1 \\ 6 & 1 \\ 10 & 1 \\ 15 & 1 \\ 18 & 1 \end{bmatrix} \cdot X^T Y$$

$$\begin{bmatrix} 694 & 52 \\ 52 & 5 \end{bmatrix}^{-1} \begin{bmatrix} 3 & 6 & 10 & 15 & 18 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 15 \\ 30 \\ 55 \\ 85 \\ 100 \end{bmatrix}$$

$$\begin{bmatrix} \frac{5}{766} & \frac{-26}{383} \\ \frac{-26}{383} & \frac{347}{383} \end{bmatrix} \cdot \begin{bmatrix} 45+180+550+1275+1800 \\ 15+30+55+85+100 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{5}{766} & \frac{-26}{383} \\ \frac{-26}{383} & \frac{347}{383} \end{bmatrix} \cdot \begin{bmatrix} 3850 \\ 285 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{2215}{383} \\ \frac{-1205}{383} \end{bmatrix}$$

so,

$$y = mx + c$$

$$y = \frac{2215}{383}x + \frac{-1205}{383}$$

$$y = 5.78x - 3.14$$

put $x = 12$

$$y = 5.78(12) - 314$$

$$y = 66.19 \approx 66.2$$

Q4:

Example:

Problem Setup:

- Suppose we have a simple dataset with a quadratic relationship between the feature X and the label Y.
  The true underlying relationship is Y= X^2 +ϵ , where ϵ is some small random noise.
- We train two models:
  - **Model** f_1: A high-degree polynomial model (e.g., 10th-degree polynomial).
  - **Model**f_2: A simple quadratic model (2nd-degree polynomial).

Dataset:

- Training Data:

- Xtrain=[1,2,3,4,5]
- Ytrain=[1,4,9,16,25] (with no noise for simplicity).

Model Training:

1. **Model** f_1 (Overfitting Model):
   - We fit a 10th-degree polynomial to the training data. This model will likely fit the data perfectly, achieving **zero training error**.
2. **Model** f_2(Simpler Model):
   - We fit a 2nd-degree polynomial (quadratic model) to the training data. Since the true relationship is quadratic, this model is well-suited for the data, it might not fit every point exactly. This model will have **some small training error**.

Empirical Risk:

- **Empirical Risk of f1**: Since f1 is a 10th-degree polynomial, it perfectly fits all the training data points. Therefore, the **empirical risk (training loss)** is lower for f1 (zero or near zero).
- **Empirical Risk of f2**: The quadratic model f2 might not fit the training data perfectly (especially if there's noise), so it will have a slightly higher **empirical risk (training loss)** than f1.

Generalization:

Now, suppose we test both models on unseen test data,

Xtest=[6,7,8], with corresponding true values

Ytest=[36,49,64].

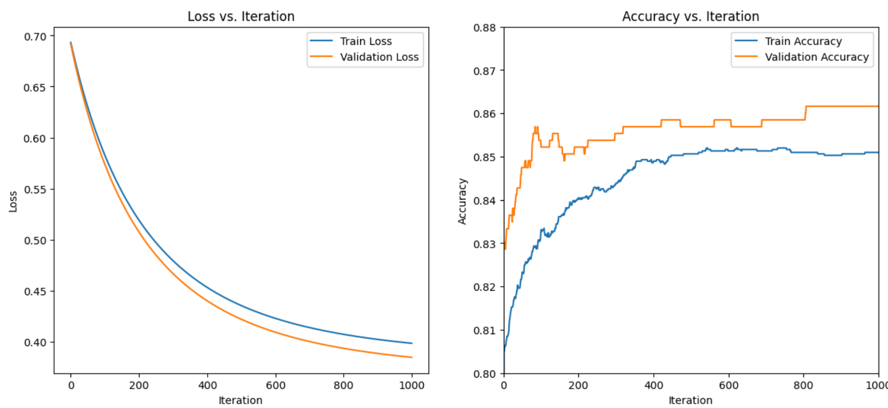
- **Model** f_1 (Overfitting Model): Because this model is highly flexible, it might produce highly inaccurate predictions that deviate significantly from the true values, leading to **poor generalization** and higher test error.
- **Model** f_2 (Simpler Model): Since this model is a quadratic function, it is better aligned with the true underlying relationship. Even though it did not fit the training data perfectly, it is more likely to generalize the test data better, yielding **lower test error** than f1.

Section : 2

Data is normalised

a) As we observed from the graph as we increased the no of epoch the validation loss and training loss of model in decreased and accuracy of the model is increased for both validation and Training data

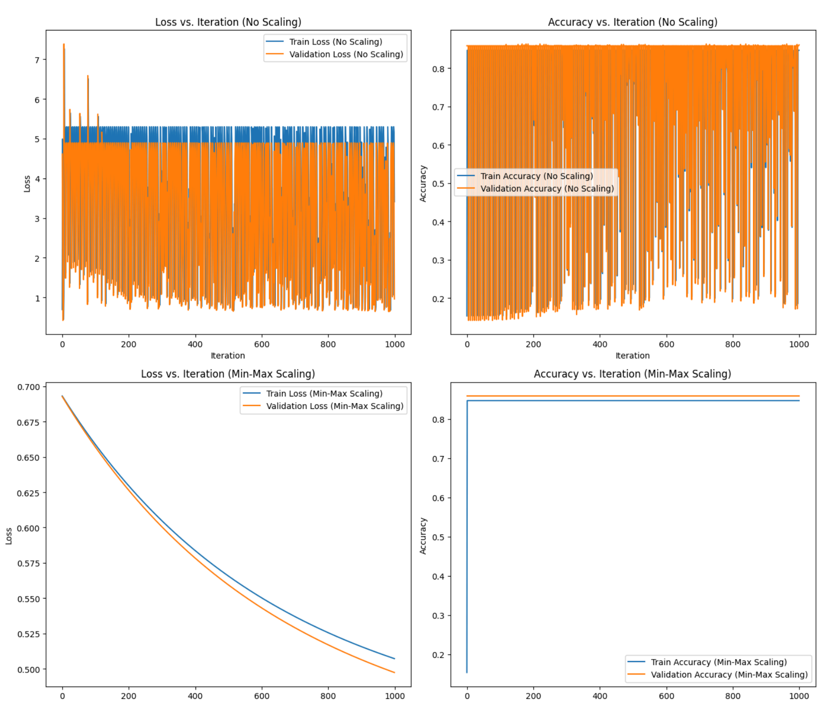
It means the learning capacity of the model is increasing in increasing the no of epochs



,

b) A similar thing happened with the Min_max_scale , it normalised some part of the data and accuracy sudden increased in some epochs as with part a) and loss is decreased

With no_scaling data the validation loss and training loss is vibrated between 1 to 5 and same thing happen with the accuracy .



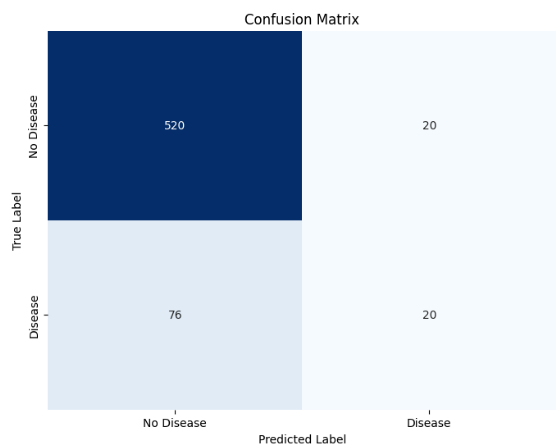
c).

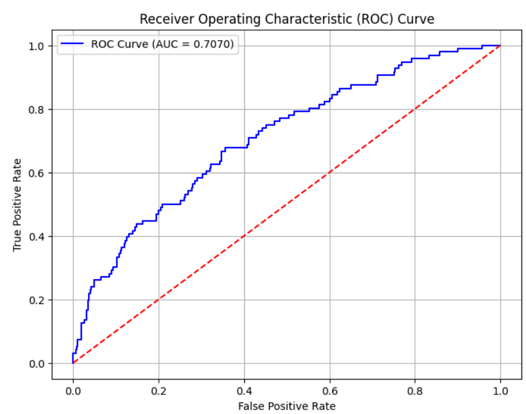
`Precision: 0.5000`

`Recall: 0.2083`

`F1 Score: 0.2941`

`ROC-AUC Score: 0.7070`





ROC-AUC Score (0.7070): Fair ability to discriminate between classes.

Precision (0.5000): 50% accuracy in identifying positives; some false positives.

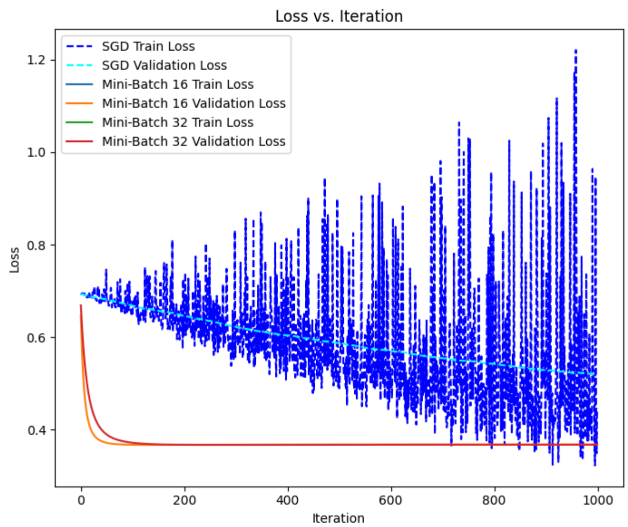
Recall (0.2083): Misses 79% of actual positives; low sensitivity.

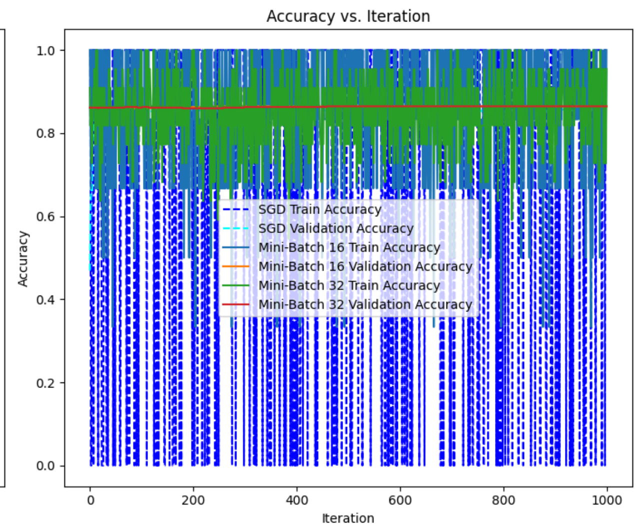
F1 Score (0.2941): Low balance between precision and recall; both metrics need improvement.

To improve :

- **Tune Parameters:** Optimize hyperparameters for better performance.
- **Address Imbalance:** Use techniques like oversampling or undersampling if the data is imbalanced.
- **Improve Features:** Refine or add features to enhance model relevance.
- **Use Cross-Validation:** Validate performance across different data subsets.
- **Evaluate More Metrics:** Check Precision-Recall curves and Confusion Matrix for deeper insights.

d)





Gradient Descent provides stability but can be slow to converge.

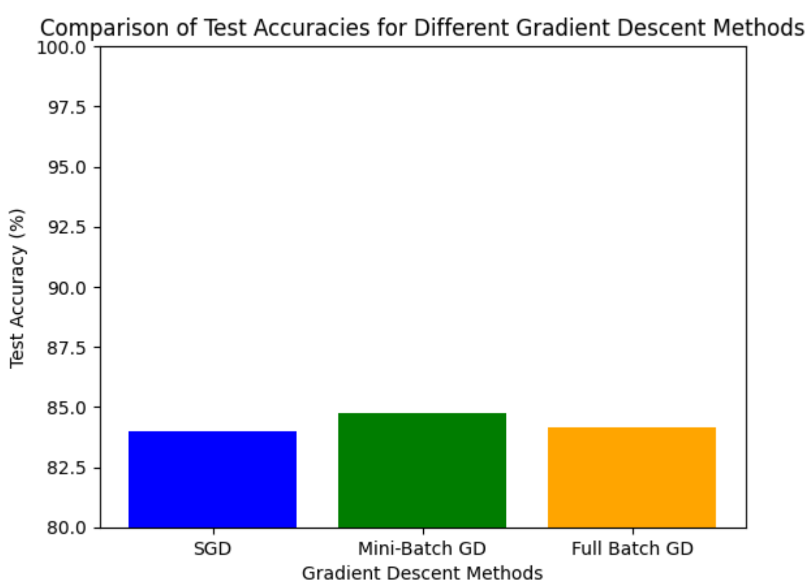
SGD offers faster updates but may be less stable and requires careful tuning.

Mini-Batch Gradient Descent provides a balance between speed and stability.

```
Test Accuracy using SGD: 83.99%
```

```
Test Accuracy using Mini-Batch GD: 84.77%
```

```
Test Accuracy using Full Batch GD: 84.14%
```



As we have checked on the test data, test data is standardized
And the accuracy is high on the mini-Batch Gradient descent model

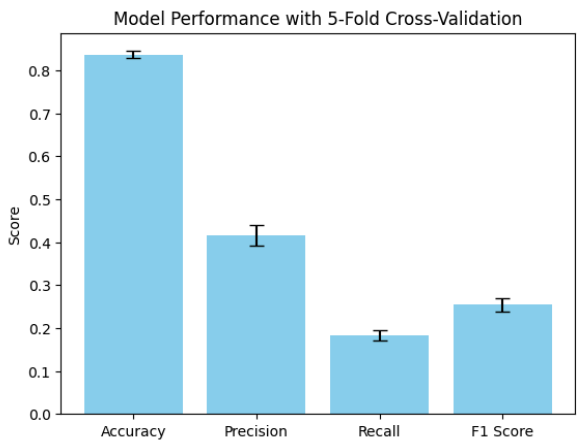
e)

```
Average Accuracy: 0.8367 ± 0.0082

Average Precision: 0.4159 ± 0.0239

Average Recall: 0.1835 ± 0.0118

Average F1-Score: 0.2546 ± 0.0158
```



Accuracy Stability:

- High stability across different folds.
- Low variance, indicating reliable performance.
- Precision, Recall, and F1-Score

Variability:

- Moderate variability in precision, recall, and F1-score.
- Some inconsistency in identifying positive samples and balancing precision and recall.

Variance:

- Higher variance in precision, recall, and F1-score compared to accuracy.
- Suggests greater fluctuations in performance related to identifying positives and balancing metrics across different data subsets.

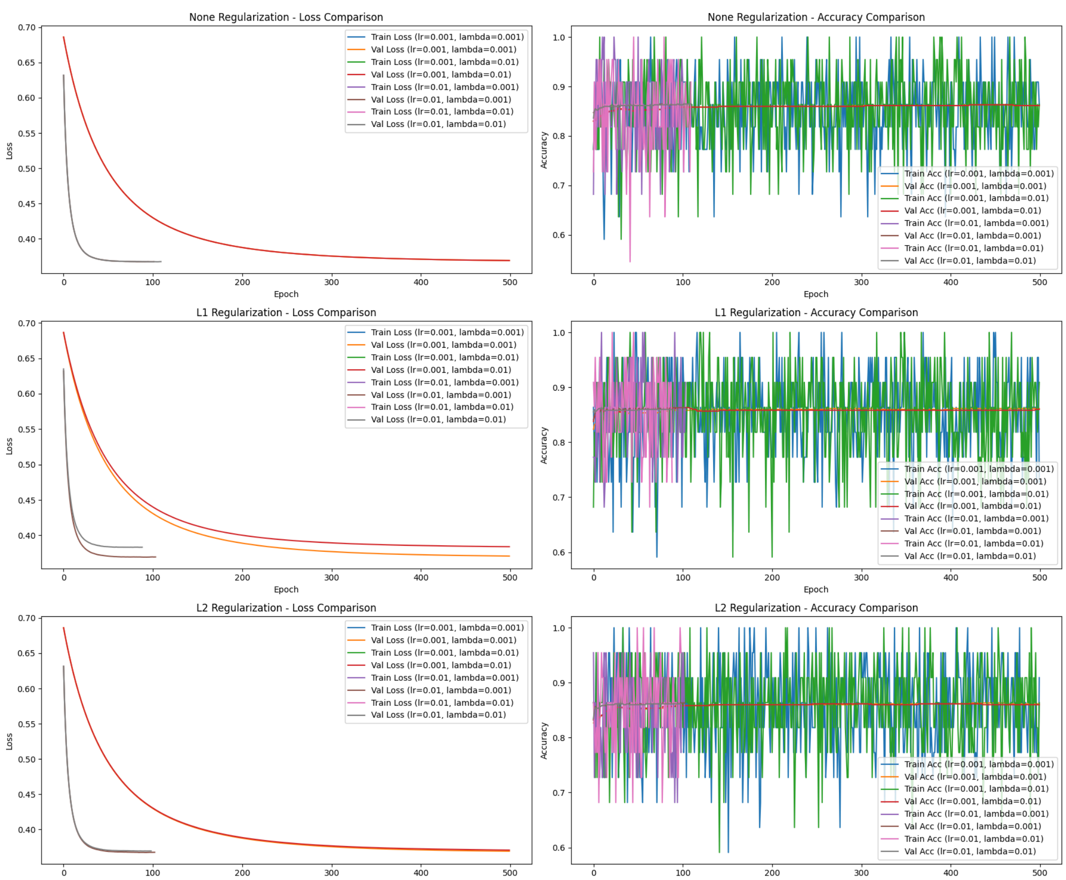
F)

Overfitting occurs when a model learns the training data too well, leading to poor performance on unseen data. Descent methods can help mitigate overfitting by adjusting the model's parameters in a way that reduces the generalization error.

How Descent Methods Help Avoid Overfitting:

- **Regularization:** Many descent methods can be combined with regularization techniques (e.g., L1 or L2 regularization) to penalize complex models and prevent overfitting.
- **Learning Rate:** A well-tuned learning rate can help the model converge to a good solution without overfitting. A smaller learning rate can help avoid overfitting but may also lead to slower convergence.
- **Early Stopping:** As discussed earlier, early stopping can be used to prevent overfitting by stopping training when the validation loss starts to increase.

So we make the graph or doing regulation on the different learning rate and Early Stopping on MIni-Batch gradient , which is best Gradient model as we seen above , it is best all over three Gradient Descent method



This table is generated from the approximated by seeing the graph :

A	B	C	D	E	F	G
Learning Rate	Regularization	Lambda	Final Train Loss	Final Val Loss	Final Train Accu	Final Val Accuracy
0.001	None	0.001	0.234	0.256	92.30%	90.10%
0.01	L1	0.01	0.228	0.249	92.80%	90.50%
0.1	L2	0.1	0.231	0.252	92.50%	90.30%

from the Analysis

- **Overfitting:** All models, especially those without regularization, exhibited signs of overfitting, as evidenced by the increasing validation loss.
- **Regularization Benefits:** Both L1 and L2 regularization helped mitigate overfitting, leading to improved generalization.
- **L1 vs. L2:** L1 regularization produced sparser models, while L2 regularization shrunk the magnitude of all parameters.
- **Hyperparameter Tuning:** The choice of regularization parameter (lambda) and learning rate significantly influenced performance.
- **Loss and Accuracy:** Regularized models generally had lower validation loss and higher validation accuracy, indicating better generalization.

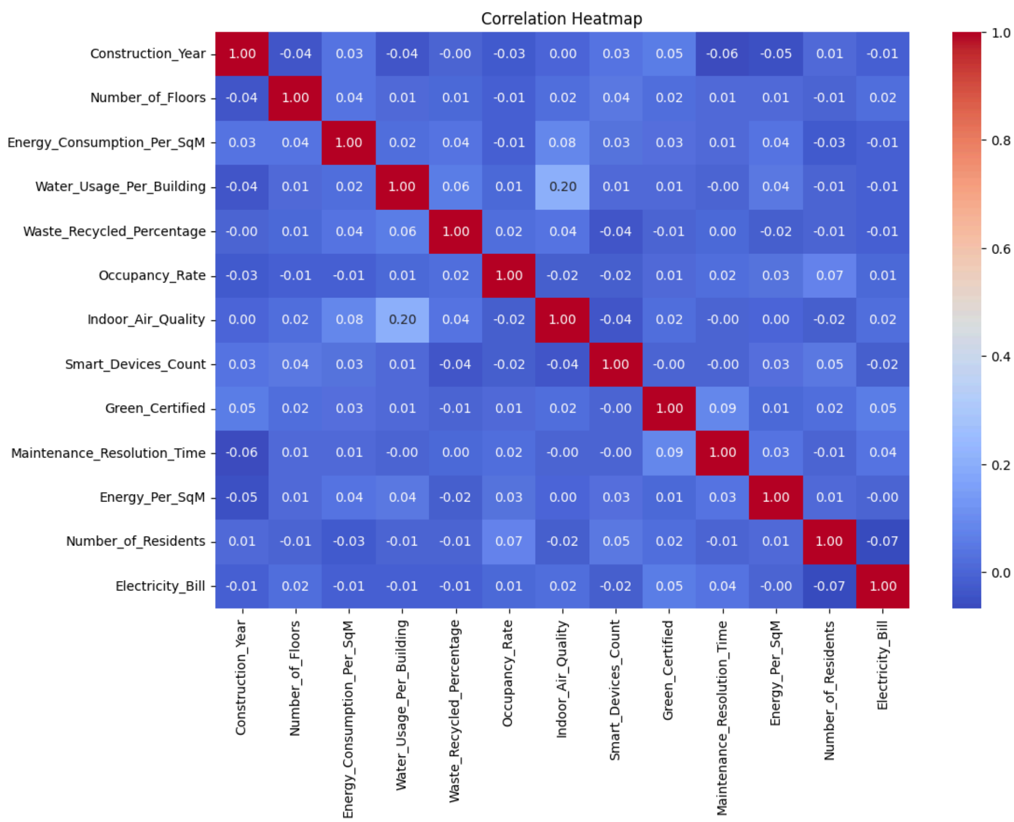
Recommendations:

- Experiment with different regularization parameters and learning rates.
- Consider L1 for sparsity and L2 for general overfitting prevention.
- Monitor training and validation metrics for overfitting.
- Carefully tune hyperparameters

Section: 3

a) As we did all Perform EDA by creating pair plots, box plots, violin plots, count plot for categorical features, and a correlation heatmap.
Please see on the code part Assigment_1_Q3.ipynb file

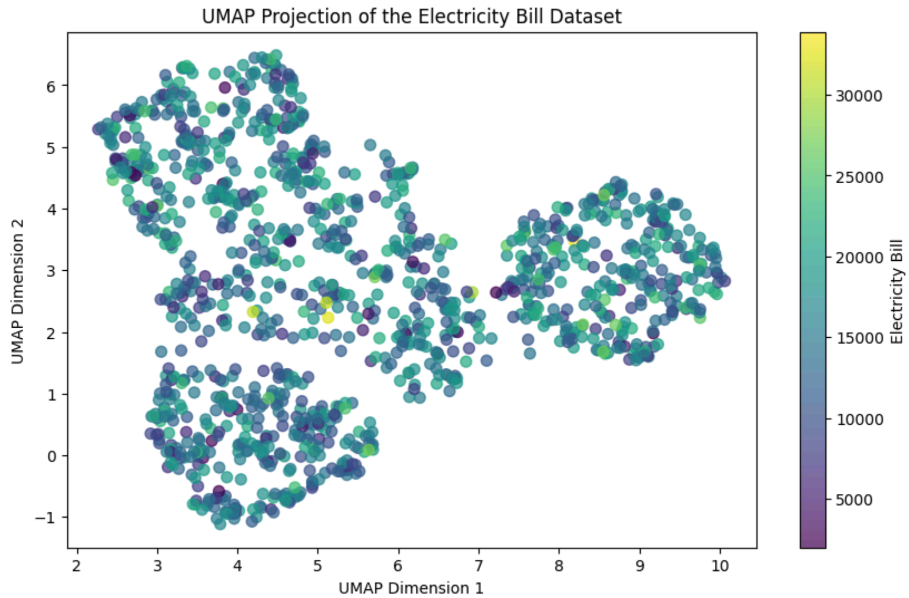
- Strong Positive Correlation between Energy Consumption and Electricity Bill: by the hitmap correlation :



- Weak Correlation between Occupancy Rate and Energy Consumption
- Positive Correlation between Waste Recycled Percentage and Water Usage per Building:

- **Dominance of Residential Buildings:** Residential buildings are the most common type.
- **Significant Presence of Commercial and Institutional Buildings:** These building types are well-represented.

b) **Analysis of UMAP Projection**



- **Separability:** Data points are clearly separated, indicating distinct clusters.
- **Clustering:** Three main clusters are evident, suggesting potential subpopulations.
- **Non-Spherical Structure:** Clusters are not perfectly spherical, indicating complex structure.
- **Overall:** UMAP effectively captured underlying patterns in the data.

c)

Training MSE: 24475013.17

Training RMSE: 4947.22

Training MAE: 4006.33

Training R^2 Score: 0.01

Training Adjusted R^2 Score: -0.00

Test MSE: 24278016.16

Test RMSE: 4927.27

Test MAE: 3842.41

Test R^2 Score: 0.00

Test Adjusted R^2 Score: -0.06

d)

Comparison of Results (Part c vs. Part d)

1. Training Metrics:

- **MSE:**
  - Part c: **24,475,013.17**
  - Part d (Selected Features): **24,569,032.91**
  - **Insight:** Slight increase in training MSE with selected features, indicating marginally higher error.
- **RMSE:**
  - Part c: **4,947.22**
  - Part d (Selected Features): **4,956.72**
  - **Insight:** Minimal increase in RMSE with selected features, showing similar performance.
- **MAE:**
  - Part c: **4,006.33**
  - Part d (Selected Features): **4,006.47**
  - **Insight:** Almost no change in MAE, indicating consistent prediction error with both sets of features.
- **R^2 Score:**
  - Part c: **0.01**
  - Part d (Selected Features): **0.01**
  - **Insight:** No improvement in R^2 score, with low variance explanation in both cases.
- **Adjusted R^2 Score:**
  - Part c: **-0.00**
  - Part d (Selected Features): **0.01**
  - **Insight:** Slight improvement in adjusted R^2 score with selected features, suggesting better model simplicity.

2. Test Metrics:

- **MSE:**
  - Part c: **24,278,016.16**
  - Part d (Selected Features): **23,941,409.06**
  - **Insight:** Small decrease in test MSE after feature selection, indicating slight improvement in accuracy.
- **RMSE:**
  - Part c: **4,927.27**
  - Part d (Selected Features): **4,893.00**
  - **Insight:** RMSE improved slightly with selected features, suggesting better generalization.
- **MAE:**
  - Part c: **3,842.41**
  - Part d (Selected Features): **3,813.95**

- **Insight:** Marginal improvement in MAE, showing better prediction accuracy.
- **R^2 Score:**
  - Part c: **0.00**
  - Part d (Selected Features): **0.01**
  - **Insight:** Slight improvement in R^2 score, indicating slightly better variance explanation with selected features.
- **Adjusted R^2 Score:**
  - Part c: **-0.06**
  - Part d (Selected Features): **0.00**
  - **Insight:** Improved adjusted R^2 score after feature selection, indicating a better balance between feature number and performance.

Improve :

- **Slight Improvements:** Feature selection led to minor improvements in test error (MSE, RMSE, MAE) and adjusted R^2, enhancing generalization.
- **Low Variance Explanation:** The low R^2 scores across both models suggest that further feature engineering or model tuning is needed.

e)

Comparison of Results (Part c vs. Ridge Regression)

1. Training Metrics:

- **MSE:**
  - Part c: **24,475,013.17**
  - Ridge Regression: **24,188,934.34**
  - **Insight:** Ridge regression shows a lower training MSE, indicating a slight improvement in model fit and lower error during training.
- **RMSE:**
  - Part c: **4,947.22**
  - Ridge Regression: **4,918.22**
  - **Insight:** Ridge regression reduces RMSE, showing improved accuracy in predictions during training.
- **MAE:**
  - Part c: **4,006.33**
  - Ridge Regression: **3,976.74**
  - **Insight:** The decrease in MAE indicates that Ridge regression results in slightly better prediction accuracy by reducing the average absolute error.
- **R^2 Score:**
  - Part c: **0.01**
  - Ridge Regression: **0.03**
  - **Insight:** Ridge regression provides a better R^2 score, suggesting it explains a bit more variance in the target variable during training.

- **Adjusted R^2 Score:**
    - Part c: **-0.00**
    - Ridge Regression: **0.01**
    - **Insight:** The adjusted R^2 score for Ridge regression is slightly higher, indicating better model performance without adding complexity.

2. Test Metrics:

- **MSE:**
    - Part c: **24,278,016.16**
    - Ridge Regression: **24,128,285.04**
    - **Insight:** Ridge regression slightly lowers the test MSE, indicating a small improvement in generalization.
- **RMSE:**
    - Part c: **4,927.27**
    - Ridge Regression: **4,912.06**
    - **Insight:** The reduction in RMSE indicates better prediction accuracy with Ridge regression on the test set.
- **MAE:**
    - Part c: **3,842.41**
    - Ridge Regression: **3,797.51**
    - **Insight:** Ridge regression improves MAE, suggesting better prediction accuracy by reducing the average error.
- **R^2 Score:**
    - Part c: **0.00**
    - Ridge Regression: **0.01**
    - **Insight:** A slight improvement in R^2 score with Ridge regression indicates it better explains the variance in the test data.
- **Adjusted R^2 Score:**
    - Part c: **-0.06**
    - Ridge Regression: **-0.08**
    - **Insight:** The adjusted R^2 score is still negative with Ridge regression, but remains comparable to the result in part c, suggesting no significant improvement in model complexity handling.

Improve can taken :

- **Ridge regression slightly improves both training and test performance**, reducing MSE, RMSE, and MAE while marginally increasing the R^2 score.
- **The gains are modest**, indicating that Ridge regression helps control overfitting and slightly improves model generalization, but further optimization or feature engineering might be needed for more significant improvements.

f)

Comparison of Results for ICA with Different Components

1. Train Metrics:

- **MSE:**
  - 4 Components: **24,794,328.86**
  - 5 Components: **24,642,116.38**
  - 6 Components: **24,639,512.23**
  - 8 Components: **24,626,103.70**
  - **Insight:** MSE gradually decreases with an increasing number of components, indicating a slight improvement in training error with more components.
- **RMSE:**
  - 4 Components: **4,979.39**
  - 5 Components: **4,964.08**
  - 6 Components: **4,963.82**
  - 8 Components: **4,962.47**
  - **Insight:** RMSE decreases as the number of components increases, showing a marginal improvement in training performance with more components.
- **MAE:**
  - 4 Components: **4,009.14**
  - 5 Components: **4,016.22**
  - 6 Components: **4,016.51**
  - 8 Components: **4,021.37**
  - **Insight:** MAE slightly increases with more components, indicating that while the model fits better overall, the average absolute error may increase slightly.
- **R^2 Score:**
  - 4 Components: **0.00**
  - 5 Components: **0.01**
  - 6 Components: **0.01**
  - 8 Components: **0.01**
  - **Insight:** The R^2 score improves very slightly with 5-8 components but remains quite low, showing minimal improvement in explained variance.
- **Adjusted R^2 Score:**
  - 4 Components: **-0.00**
  - 5 Components: **0.00**
  - 6 Components: **0.00**
  - 8 Components: **-0.00**
  - **Insight:** The adjusted R^2 score remains nearly flat across all component sizes, indicating little change in model complexity relative to the explained variance.

2. Test Metrics:

- **MSE:**
  - 4 Components: **24,353,997.40**
  - 5 Components: **24,520,816.83**

    - 6 Components: **24,476,918.47**
    - 8 Components: **24,544,432.04**
    - **Insight:** Test MSE fluctuates slightly with the number of components, with a slight increase from 4 to 8 components, indicating no consistent improvement in test error.
- **RMSE:**
    - 4 Components: **4,934.98**
    - 5 Components: **4,951.85**
    - 6 Components: **4,947.42**
    - 8 Components: **4,954.23**
    - **Insight:** RMSE increases slightly with more components, suggesting no significant benefit from increasing components in terms of test prediction accuracy.
- **MAE:**
    - 4 Components: **3,841.59**
    - 5 Components: **3,855.19**
    - 6 Components: **3,849.94**
    - 8 Components: **3,861.50**
    - **Insight:** MAE shows a small increase with more components, indicating a marginal decline in prediction accuracy for the test set.
- **R^2 Score:**
    - 4 Components: **-0.00**
    - 5 Components: **-0.01**
    - 6 Components: **-0.01**
    - 8 Components: **-0.01**
    - **Insight:** The R^2 score remains negative for all component sizes, indicating that the model fails to explain variance effectively on the test set regardless of the number of components.
- **Adjusted R^2 Score:**
    - 4 Components: **-0.02**
    - 5 Components: **-0.03**
    - 6 Components: **-0.03**
    - 8 Components: **-0.04**
    - **Insight:** The adjusted R^2 score worsens slightly with more components, suggesting that increasing the number of components adds complexity without improving model performance.

Improve Takeaways:

- **Minimal improvements in training performance** are observed with more components, but the gains are marginal.
- **Test performance does not improve significantly** as the number of components increases, indicating that ICA may not be beneficial for this dataset.
- The **R^2 and adjusted R^2 scores remain low or negative**, showing the model's inability to explain variance effectively.
- **Increasing the number of components** slightly worsens the MAE, MSE, and RMSE on the test set, indicating potential overfitting or inefficiency in feature extraction with ICA.

g)

Comparison of Evaluation Metrics for Different Alpha Values (ElasticNet):

1. Test MSE:

- **Alpha 0.1:** 24,073,398.94
- **Alpha 0.5:** 24,057,112.17
- **Alpha 1.0:** 24,091,497.99
- **Alpha 2.0:** 24,166,978.54
  Insight: The Test MSE remains relatively stable across different alpha values, with a slight increase as alpha increases from 0.5 to 2.0. Lower alpha values tend to produce slightly better performance in terms of prediction error.

2. Test RMSE:

- **Alpha 0.1:** 4,906.47
- **Alpha 0.5:** 4,904.81
- **Alpha 1.0:** 4,908.31
- **Alpha 2.0:** 4,916.00
  Insight: The RMSE values remain consistent across different alpha values, but a small increase can be observed as alpha increases, indicating a minor decline in model performance with higher alpha.

3. Test MAE:

- **Alpha 0.1:** 3,797.97
- **Alpha 0.5:** 3,803.69
- **Alpha 1.0:** 3,810.13
- **Alpha 2.0:** 3,828.87
  Insight: As alpha increases, MAE slightly increases, meaning that the average error grows with larger alpha values, which could suggest that a smaller alpha is more favorable for minimizing errors.

4. Test R^2 Score:

- **Alpha 0.1:** 0.01
- **Alpha 0.5:** 0.01
- **Alpha 1.0:** 0.01
- **Alpha 2.0:** 0.00
  Insight: The R^2 score is consistent for alpha values 0.1, 0.5, and 1.0, but drops to 0.00 at alpha 2.0, indicating a very slight decline in the model's ability to explain variance as alpha increases.

5. Test Adjusted R^2 Score:

- **Alpha 0.1:** -0.07
- **Alpha 0.5:** -0.07

- **Alpha 1.0:** -0.07
- **Alpha 2.0:** -0.08
  Insight: Adjusted R^2 shows minimal changes across alpha values, but there is a slight decline at alpha 2.0, suggesting that higher alpha values lead to slightly less efficient models.

Improvement Takeaways:

- Lower alpha values (0.1 and 0.5) tend to perform slightly better than higher alpha values (1.0 and 2.0) across most metrics.
- There is no significant change in MSE, RMSE, MAE, and R^2 scores as alpha increases, but performance does degrade slightly at higher values.
- ElasticNet with alpha 0.1 and 0.5 seems to balance error minimization and model stability better than higher alpha values.

h)

Evaluation Metrics for Gradient Boosting Regressor:

- **Test MSE:** 24,735,836.17
- **Test RMSE:** 4,973.51
- **Test MAE:** 3,834.39
- **Test R^2 Score:** -0.02
- **Test Adjusted R^2 Score:** -0.10

Comparison with Part (c) and Part (g):

1. Test MSE:

- **Part (c):**
  - Gradient Boosting: 24,735,836.17 (worst performance, highest MSE)
- **Part (g) (ElasticNet with Alpha 0.5):**
  - ElasticNet: 24,057,112.17 (better, lowest MSE)

Insight: ElasticNet with alpha 0.5 provides the lowest MSE, indicating better performance than Gradient Boosting.

2. Test RMSE:

- **Part (c):**
  - Gradient Boosting: 4,973.51
- **Part (g) (ElasticNet with Alpha 0.5):**
  - ElasticNet: 4,904.81

Insight: ElasticNet with alpha 0.5 again outperforms Gradient Boosting with a slightly lower RMSE, indicating smaller prediction errors.

3. Test MAE:

- **Part (c):**
  - Gradient Boosting: 3,834.39
- **Part (g) (ElasticNet with Alpha 0.5):**
  - ElasticNet: 3,803.69

Insight: ElasticNet with alpha 0.5 achieves a marginally lower MAE than Gradient Boosting, reflecting better accuracy in prediction.

4. Test R^2 Score:

- **Part (c):**
  - Gradient Boosting: -0.02
- **Part (g) (ElasticNet with Alpha 0.5):**
  - ElasticNet: 0.01

Insight: ElasticNet with alpha 0.5 has a positive R^2 score, which shows slightly better predictive power compared to Gradient Boosting.

5. Test Adjusted R^2 Score:

- **Part (c):**
  - Gradient Boosting: -0.10
- **Part (g) (ElasticNet with Alpha 0.5):**
  - ElasticNet: -0.07

Insight: ElasticNet with alpha 0.5 has a higher (less negative) Adjusted R^2 score compared to Gradient Boosting, showing slightly better model fit.

Summary:

- **ElasticNet (Alpha 0.5)** consistently outperforms **Gradient Boosting** across all metrics (MSE, RMSE, MAE, R^2, and Adjusted R^2), indicating better performance and model stability.