# PRML ASSIGNMENT - 2

**Name : Aditya Sahani**
**Roll No : B22CS003**
**Google Colab :**
https://colab.research.google.com/drive/1ak9s9vNHX1rT6GqewZdcM-uJ0s2W9X5U?usp=sharing

## DECISION TREE CLASSIFIER

### Task 1 : Data Preprocessing and Exploration

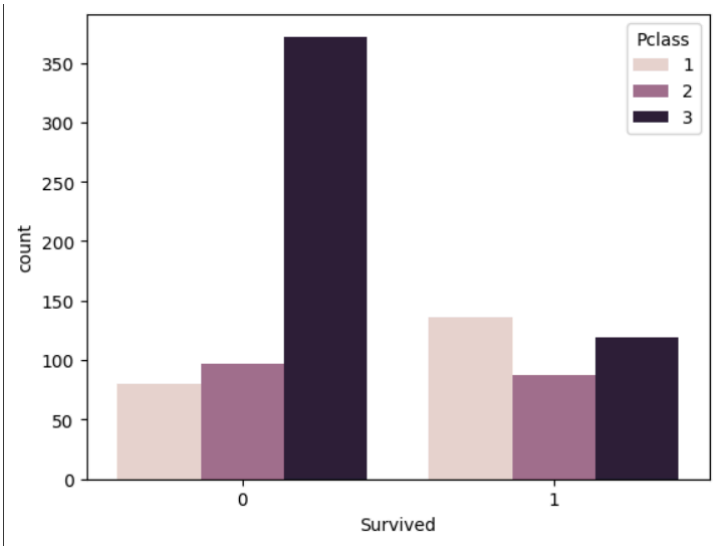
Overview of the Dataset:
- The dataset is loaded from the Titanic dataset available online.
- The dataset contains information about passengers, including their survival status (target variable), age, sex, fare, embarked, etc.

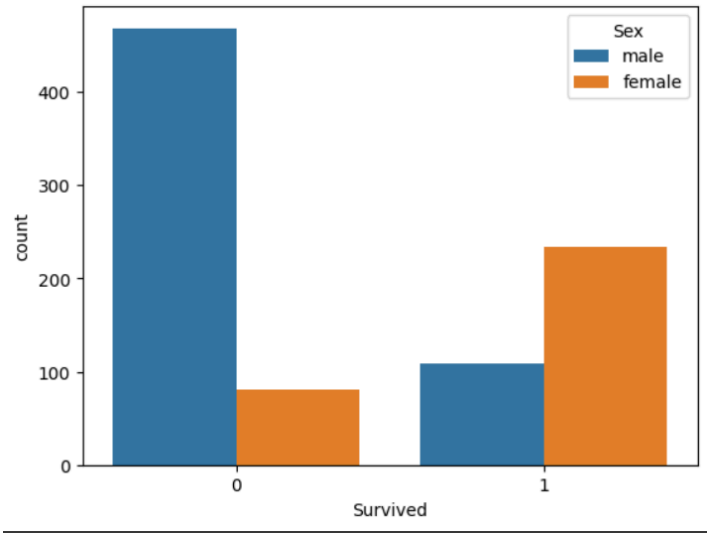
Data Exploration:
1. Checked the information about dataset columns using `df.info()`.
2. Provided an overview of the dataset using `df.describe()`.
3. Checked for null values using `df.isnull().sum()`.
4. Filled missing values in the "Embarked" column with the mode value.
5. Filled missing values in the "Age" column with the mean value.
6. Dropped the "Cabin" column due to a large number of missing values.
7. Dropped unnecessary columns like "Name," "PassengerId," and "Ticket."
8. Created a new column "Family" by combining "SibSp" and "Parch."
9. Visualized survival counts based on passenger class, gender, age groups, and embarked locations.

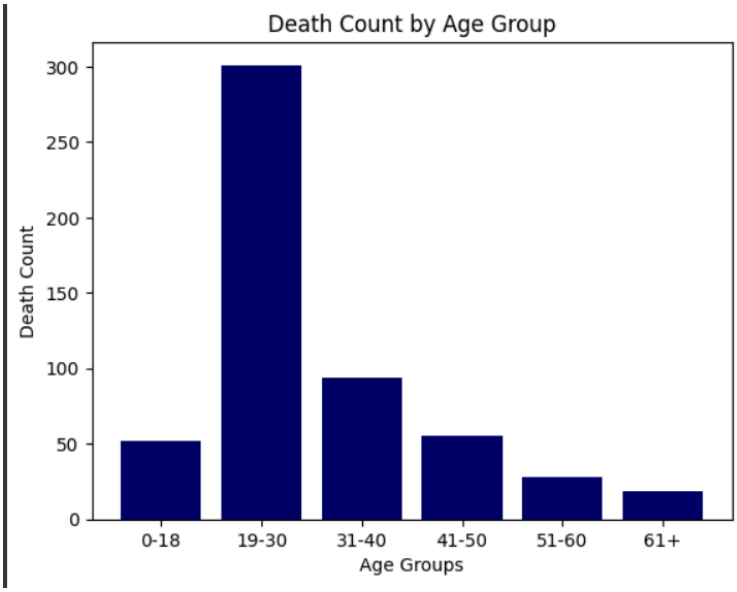
Exploratory Data Analysis:

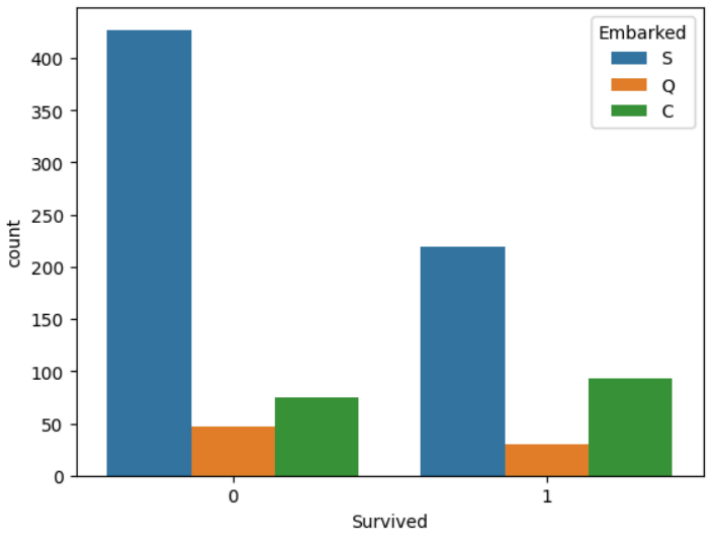
Performed various visualizations to analyze the relationships between different features and the target variable "Survived."

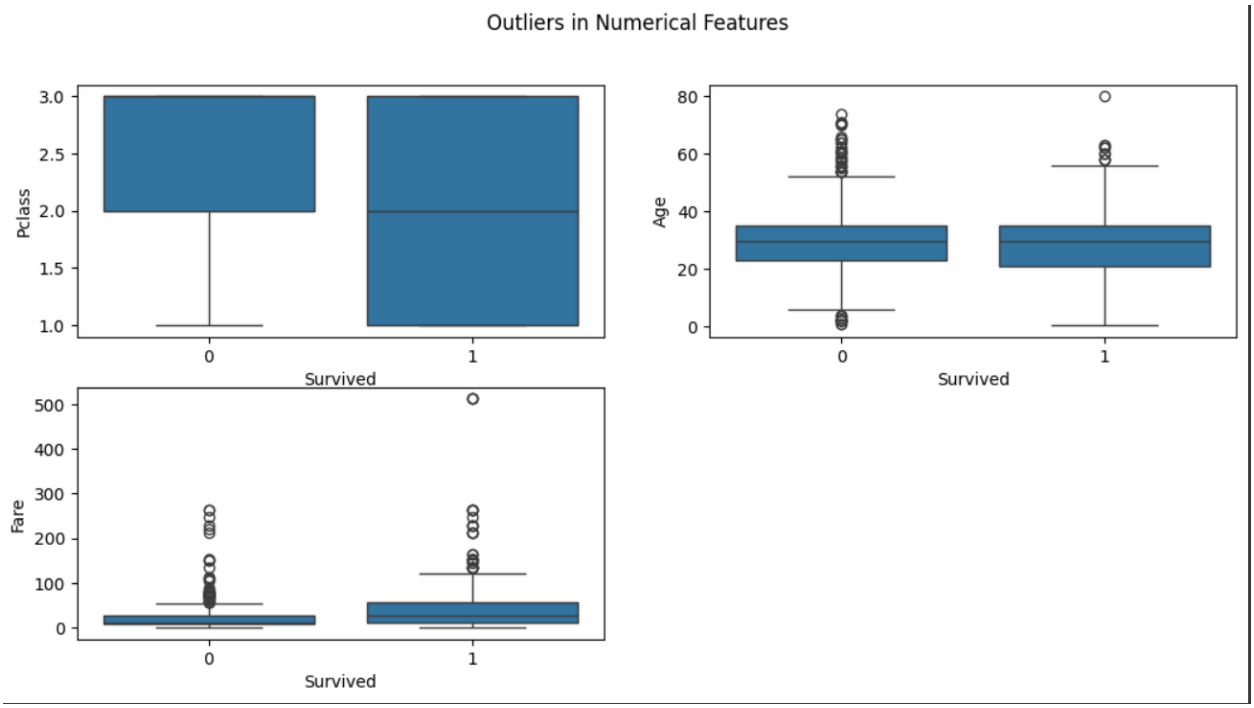
1. Count plot for survival based on passenger class.
2. Count plot for survival based on gender.
3. Bar chart for death count by age group.
4. Count plot for survival based on embarked locations.
5. Box plots and strip plots to identify outliers and visualize features.

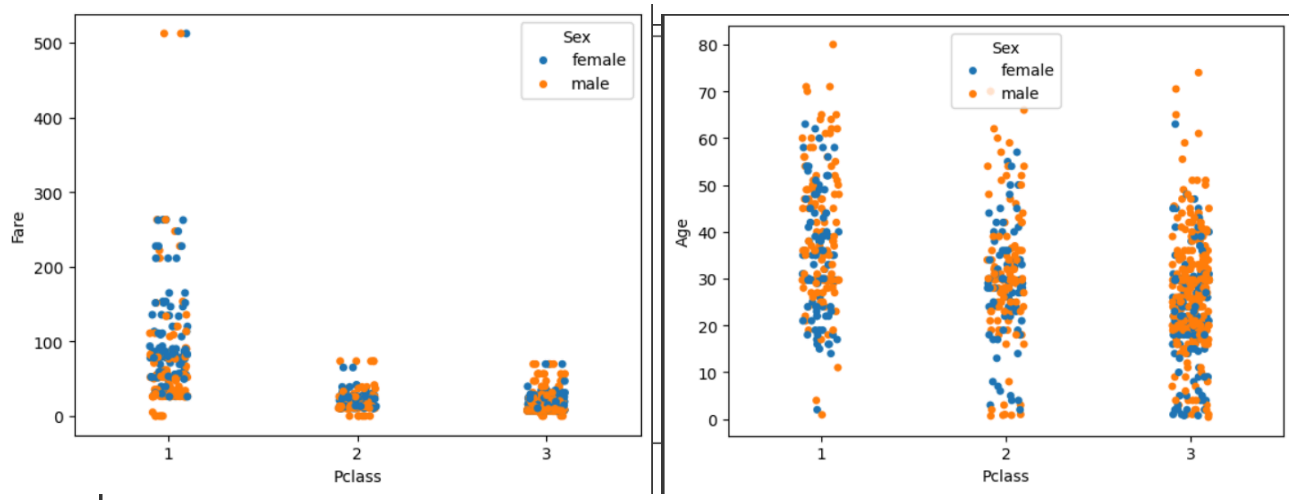





## **Ordinal and Nominal Columns:**

Identified and categorized nominal and ordinal columns in the dataset.

- Nominal Columns: "Name," "Sex," "Ticket," "Cabin," "Embarked."
- Ordinal Columns: "Pclass."

Categorical Encoding:

Defined a function `cat_encode` to perform categorical encoding on "Sex" and "Embarked" columns.

Train-Test-Split :

Split the dataset into train, validation and test in the ration 7 : 2 : 1.

**Task - 2:**

Entropy function to calculate the entropy of a particular column for performing the split.

**Task - 3:**

Performing the numerical to categorical encoding on ‘Age’ and ‘Fare’ columns because we are dealing with classification tasks.

Converting from numerical to categorical helps :

- Converting numerical features into categorical ones allows decision trees to capture non-linear relationships more effectively.
- Decision trees can be sensitive to outliers in numerical features. Converting them into categories helps mitigate the impact of outliers by grouping values into broader ranges.
- Binning numerical features can prevent decision trees from fitting the training data too closely, reducing the risk of overfitting.

**Task 4 :**

Constructing the decision tree from scratch using the ID3 algorithm. Helper functions are used to help the ID3 function for creating the nodes of the tree.

- divide_data() : used to divide the data based on information gain for further splitting.
- information_gain_func() : used to calculate the information gain of a particular feature
- select_attribute_with_highest_information_gain() : used to get the feature with highest information gain.
- majority_label() : used to get the majority class of a node.

class DecisionTree : class for training our model for our dataset . It has attributes like :

- children
- label
- label_class

**Task 5 :**

Infer() and predict() functions are defined to perform prediction on training as well as test data.

Trained two decision trees on different encoded datasets: one using mean encoding (df_mean_encoded) and the other using bin encoding (df_bin_encoded).

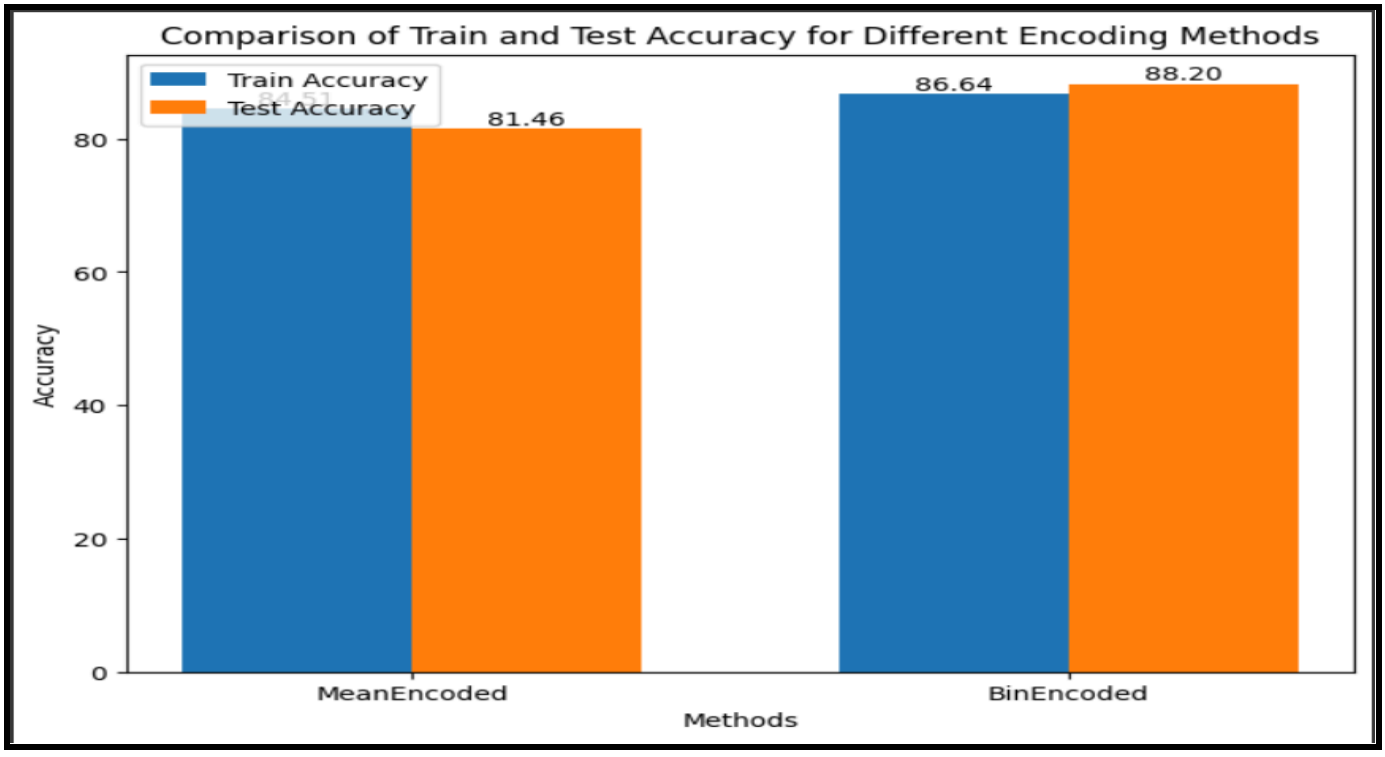
Mean Encoding : Performed encoding on a dataset using the mean-split method, where we found the best value to split the dataset into 2 classes 0 and 1.

BinEncoding : Performed encoding on a dataset using the bin method, by creating equal bins.

**Mean Encoding :**
Training Accuracy : 84.51178451178451 %
Test Accuracy : 86.64421997755332 %
Validation Accuracy : 83.1406 %

**Bin Encoding :**
Training Accuracy : 81.46067415730337 %
Test Accuracy : 88.20224719101124 %
Validation Accuracy : 86.516853 %

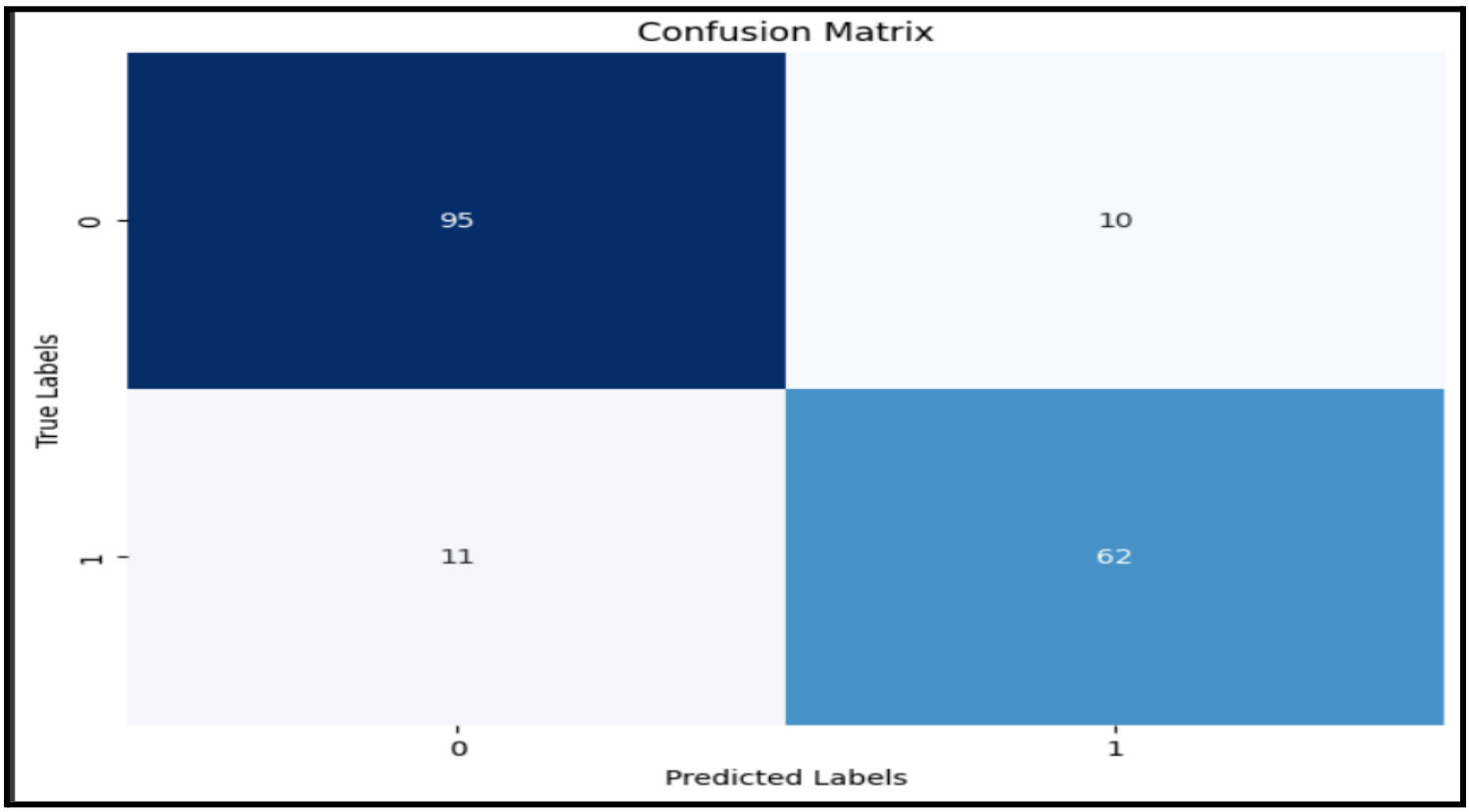
**Class-Wise Accuracy :**
Class - Wise accuracy on Training Data
{'class 0': 90.56603773584906, 'class 1': 72.22222222222221}
Class - Wise accuracy on Test Data
{'class 0': 88.67924528301887, 'class 1': 83.33333333333334}



**We can clearly see that Bin Encoding is performing better than Mean Encoding.**

**Task 6 :**

Plot the confusion matrix on the test data.




**Task 7 :**

Calculated the Precision, Recall and F1 score for the prediction.

Precision : 0.8611
Recall : 0.8493

F1_Score : 0.8552

---

## LINEAR REGRESSION - 1

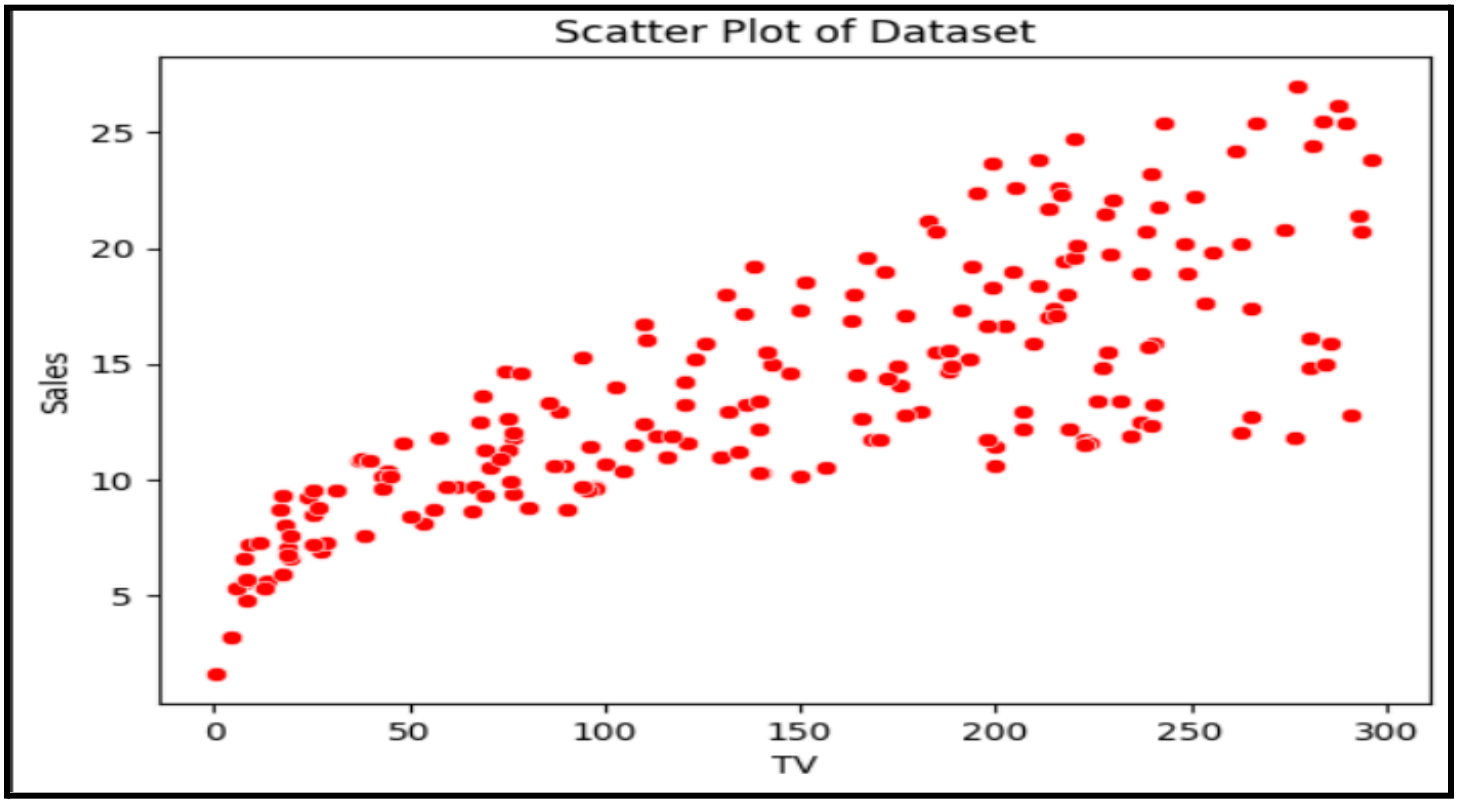
### Task 1: Data Exploration

Part (a)

The dataset's initial exploration includes displaying the first few rows using data.head().

Part (b)

A scatter plot was generated to visualize the relationship between 'TV' and 'Sales'. The dataset exhibits a linear pattern, suggesting a potential linear relationship between these two features.




Part (c)

The mean and standard deviation of 'TV' and 'Sales' were calculated and displayed, providing insights into the central tendency and spread of the data.
Mean of TV : **147.0425** Standard Deviation of TV : **85.854**
Mean of Sales : **14.0225** Standard Deviation of Sales : **5.217**

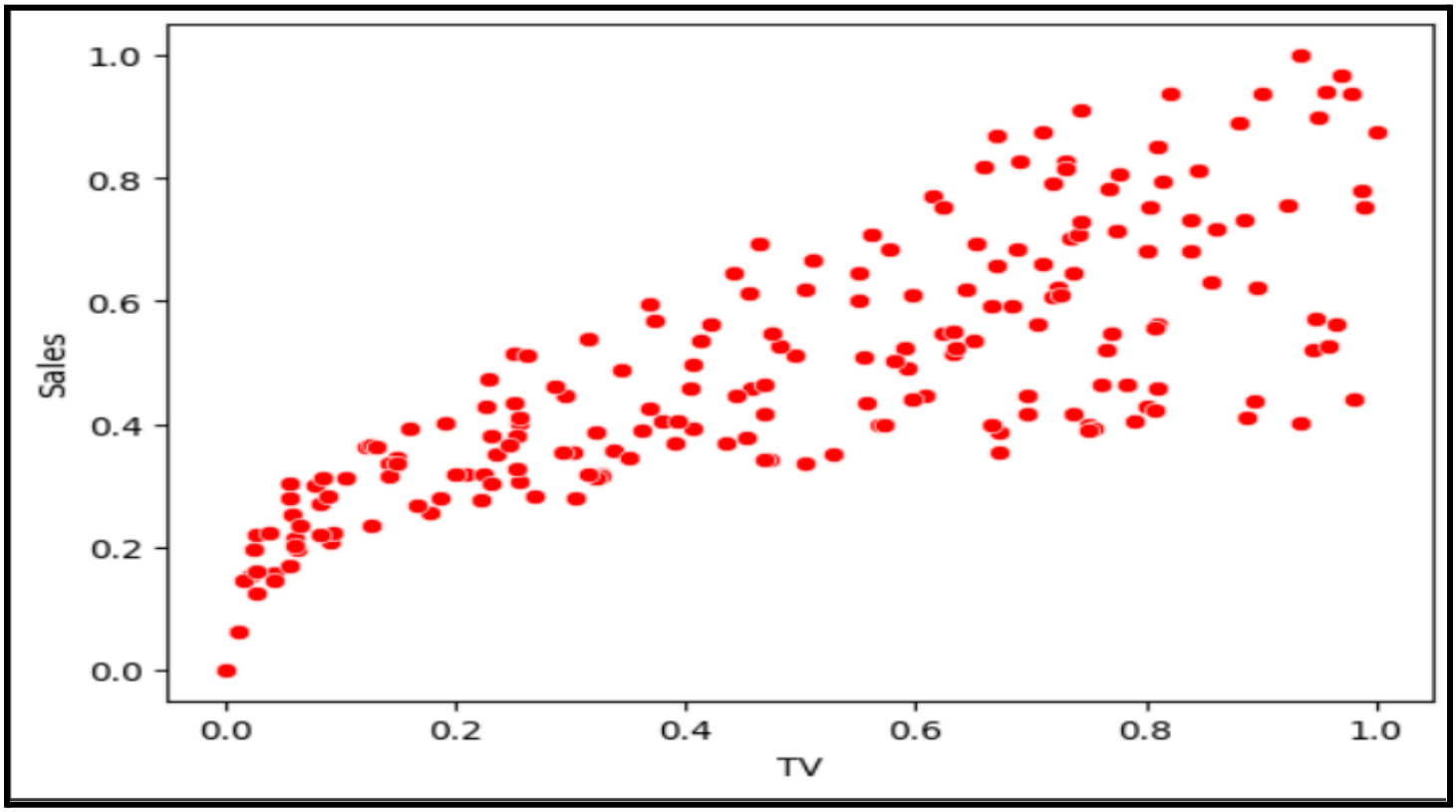
**Task 2: Data Preprocessing**

Part (a)

Missing value check was performed, revealing no missing values in the dataset.

Part (b)

MinMax Scaling normalization was applied to the dataset to bring all features to a common scale, particularly useful for gradient-based optimization algorithms.




After normalization, the nature of the plot remains the same without losing any information.

Part (c)
The dataset was split into training and testing sets using the train_test_split function from scikit-learn.

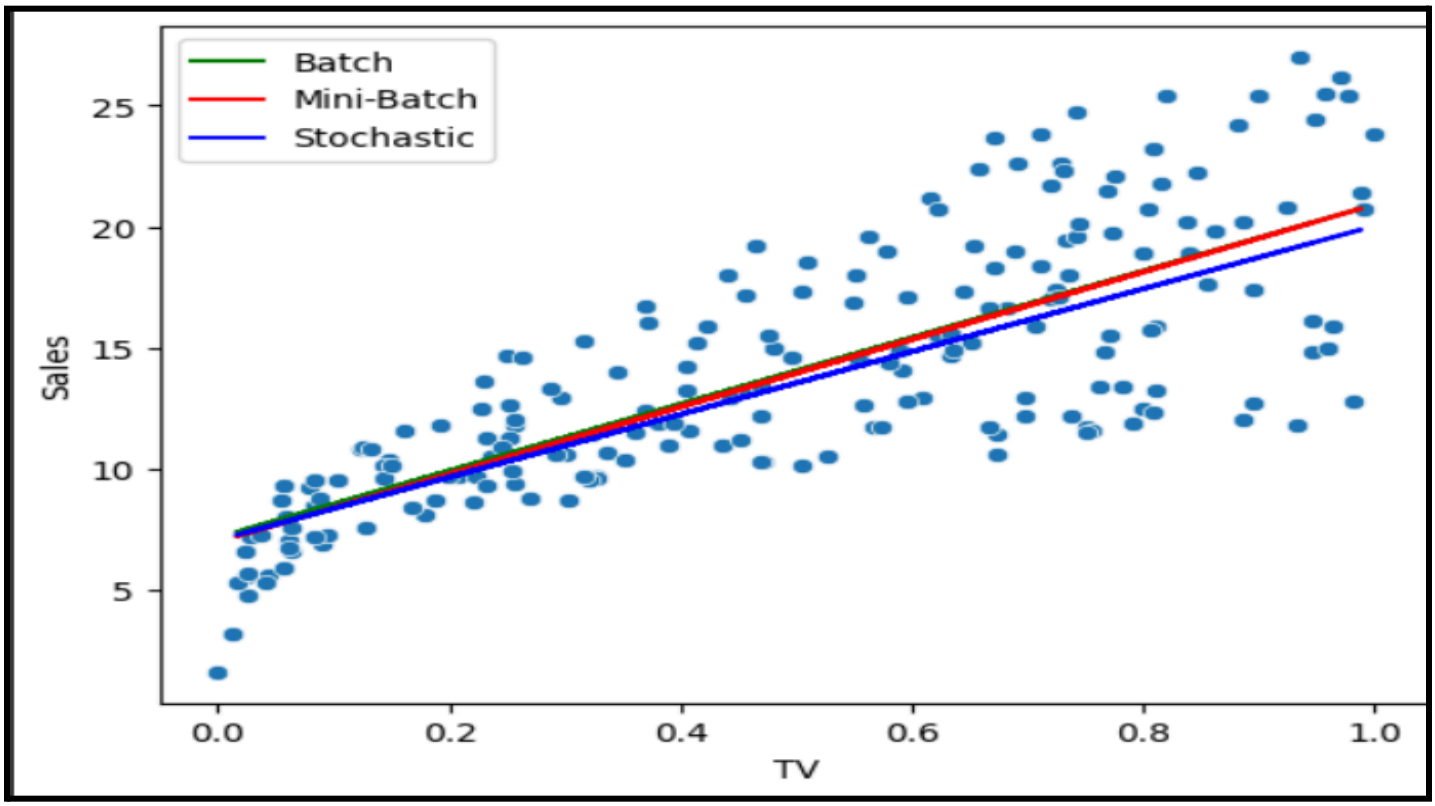
## Task 3: Linear Regression Implementation

Part (a)
Three types of Gradient Descent approaches were implemented:

- Batch Gradient Descent
- Mini-Batch Gradient Descent
- Stochastic Gradient Descent

These models were trained using the 'TV' feature to predict 'Sales'.

Part (b)
Regression lines were plotted on the scatter plot to visually compare the models' performance.

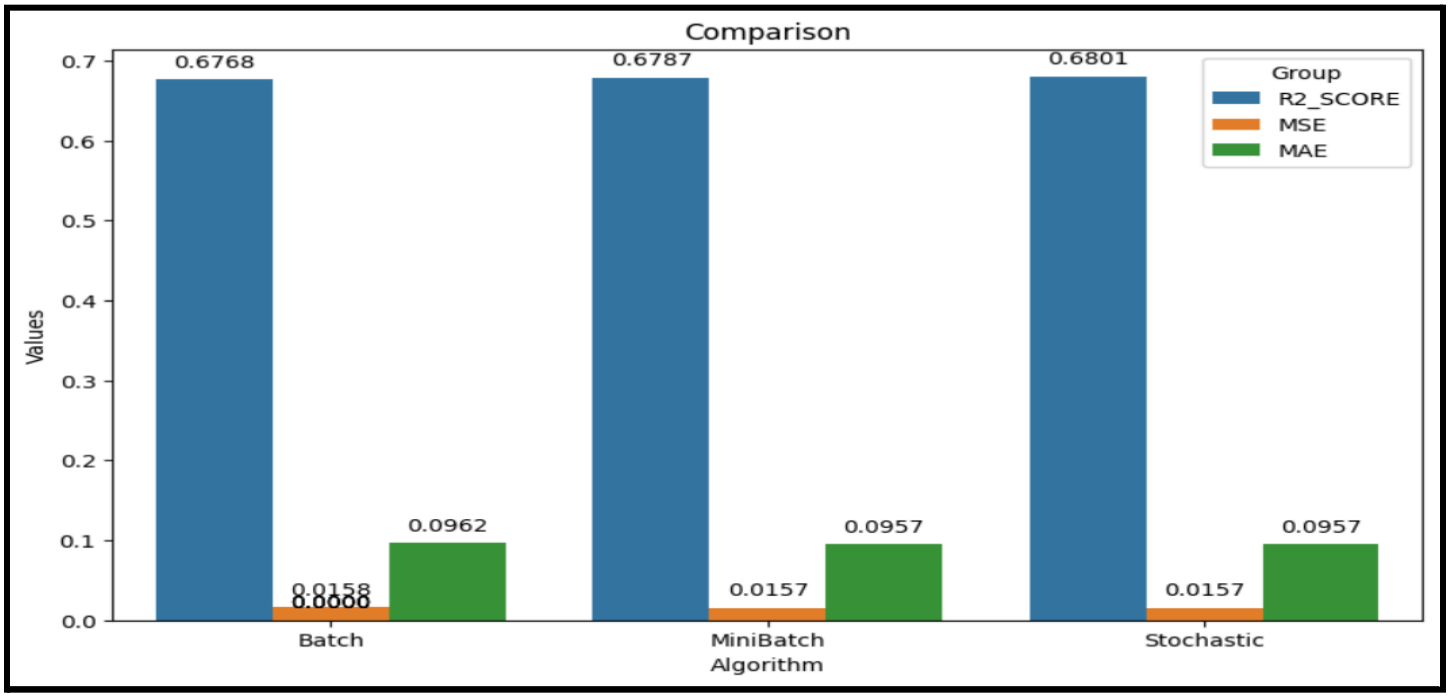



## Task 4: Evaluation

The performance of each regression model was evaluated using Mean Squared Error (MSE), Mean Absolute Error (MAE), and R2 Score. The models—Batch, Mini-Batch, and Stochastic—were compared based on these metrics to assess their effectiveness.

| NAME | R2_SCORE | MSE | MAE |
|---|---|---|---|
| Batch | 0.6766 | 0.01582 | 0.0962 |
| MiniBatch | 0.6977 | 0.0157 | 0.0957 |
| Stochastic | 0.6889 | 0.01571 | 0.0957 |

The results were presented in tabular form, and visualizations, including bar plots, were used to aid in the comparison of algorithms.



Hence, we can see that Stochastic Gradient is performing much better than other algorithms.

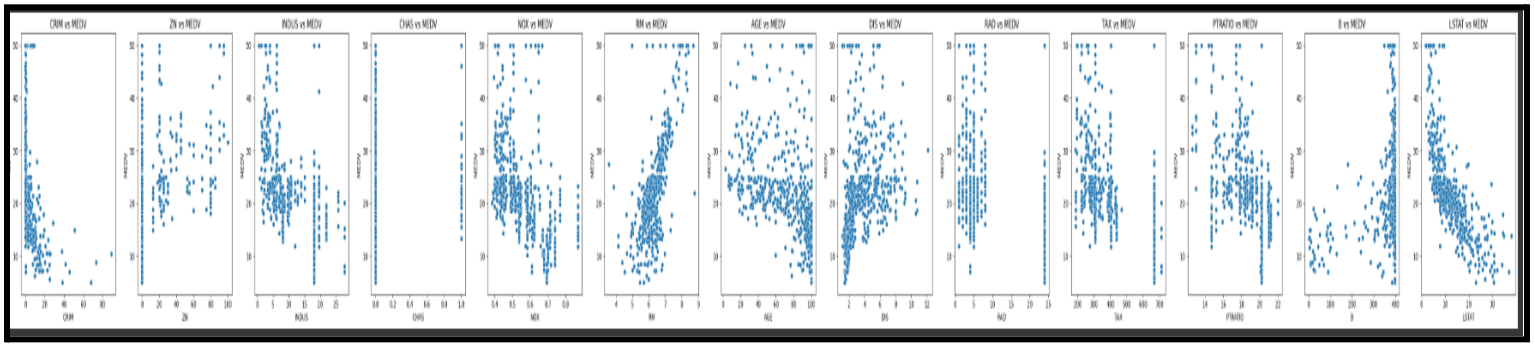
---

## LINEAR REGRESSION - 2

### Task 1 (Data Exploration)

Part (a)
The dataset loaded from the "bostonHousingData.csv" file contains information about various factors affecting housing prices in Boston. The dataset includes multiple features like crime rate, average number of rooms per dwelling, pupil-teacher ratio, and more.

Part (b)
A scatter plot matrix was created to visualize the relationship between each feature and the target variable ('MEDV' - median value of owner-occupied homes). Each scatter plot displays the relationship between a specific feature and the target variable, helping to identify potential patterns and correlations.




Part (c)
Statistical summaries, including mean and standard deviation, were computed for each feature. These summaries provide insights into the central tendency and spread of the data, aiding in the understanding of feature distributions.

**Mean of Features :**
CRIM 3.6118739711934156
ZN 11.2119341563786
INDUS 11.083991769547325
CHAS 0.06995884773662552
NOX 0.5546950592885376
RM 6.284634387351779
AGE 68.51851851851852
DIS 3.795042687747036
RAD 9.549407114624506
TAX 408.2371541501976
PTRATIO 18.455533596837945
B 356.6740316205534
LSTAT 12.715432098765433
MEDV 22.532806324110677

**Standard Deviation of Features :**
CRIM 8.720191850151599
ZN 23.388876146265478
INDUS 6.83589649864144
CHAS 0.2553404809065679
NOX 0.11587767566755595
RM 0.7026171434153233
AGE 27.99951300509237
DIS 2.105710126627611
RAD 8.707259384239366
TAX 168.53711605495903
PTRATIO 2.1649455237144406
B 91.29486438415783
LSTAT 7.155870815805251
MEDV 9.197104087379818

**Task 2 (Data Preprocessing)**

Part (a)

An initial check for missing values was performed, and it revealed no null values in the dataset. Any missing values were handled by filling them with the mean of their respective columns.

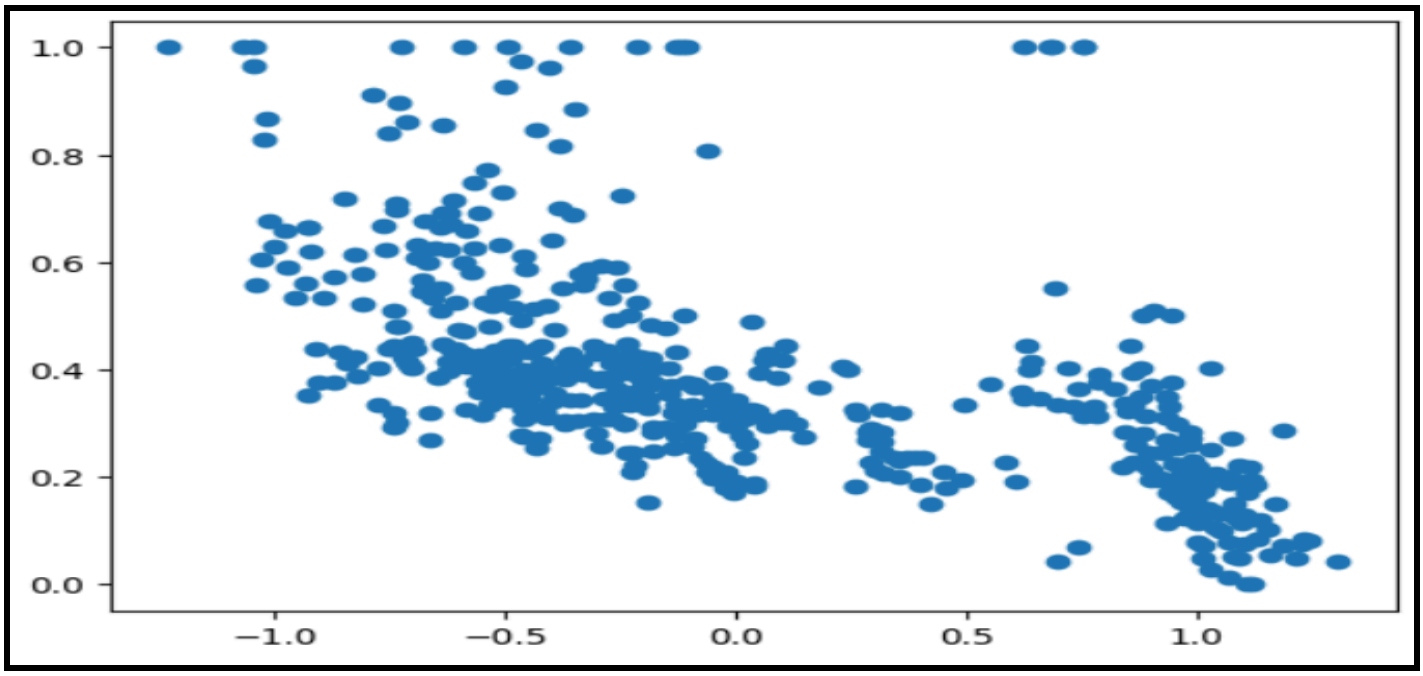
Part (b)

MinMax scaling was applied to normalize the data. This scaling method is suitable for linear regression models as it preserves the relationships between variables. Additionally, it helps to prevent features with larger scales from dominating the model.

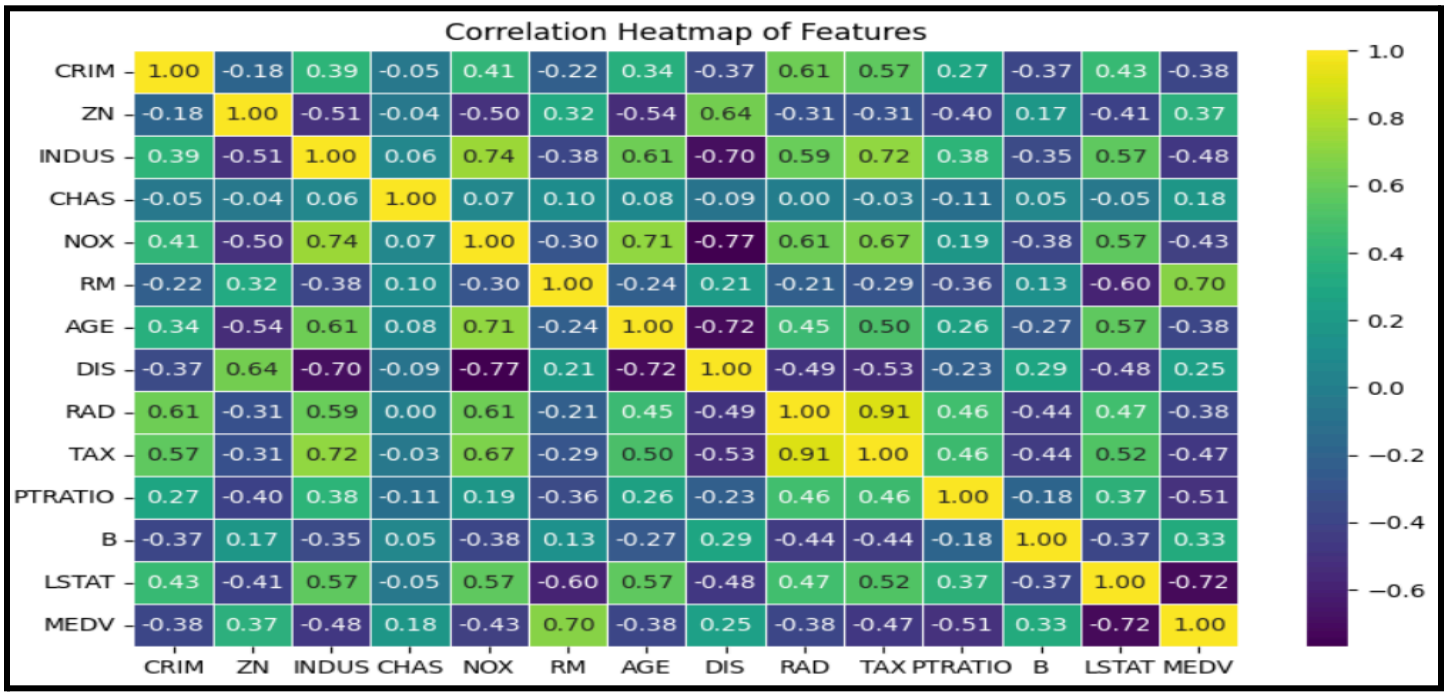
Part (c)

Principal Component Analysis (PCA) was utilized for dimensionality reduction and visualization. A scatter plot using PCA components highlighted the distribution of the target variable in the reduced feature space. A correlation heatmap was also generated to identify feature correlations.

Used PCA to reduce the data to 2 dimensions so that we can see the pattern in data. The data is almost linear so we can surely apply linear regression here.

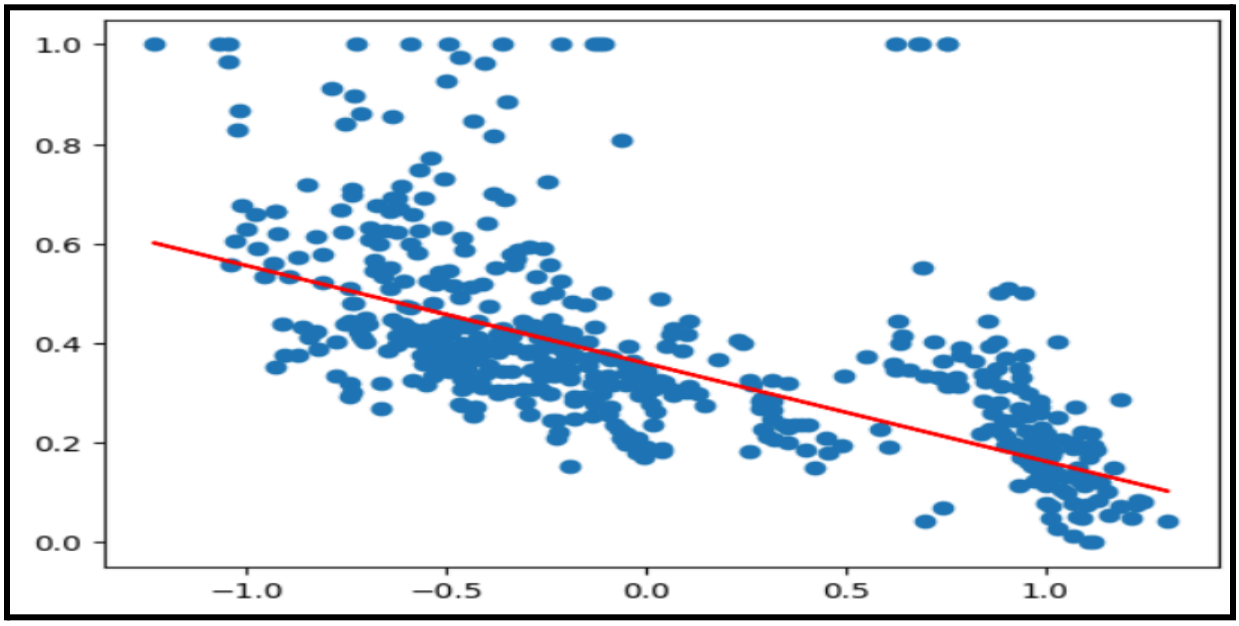



Plotted heatmap to see the correlation of features with each other.

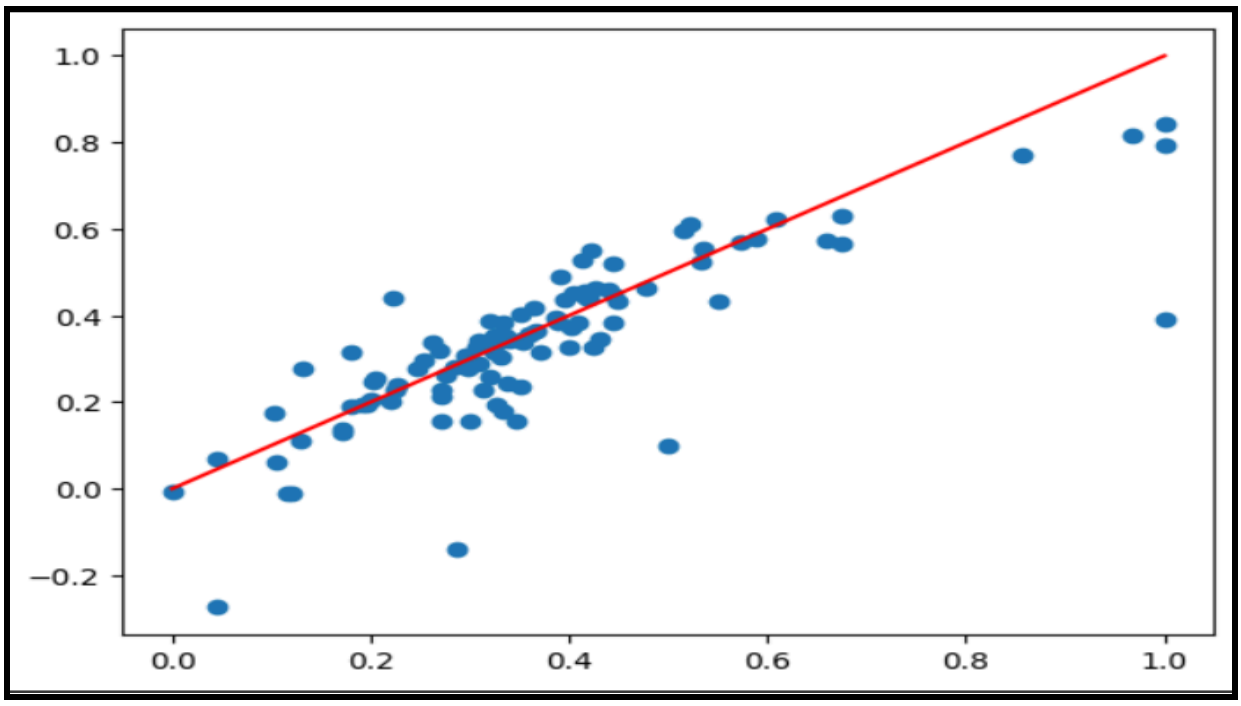



## Task 3 (Linear Regression Implementation)

Linear regression models were implemented using three different optimization algorithms: Batch Gradient Descent (BGD), Mini-Batch Gradient Descent (MBGD), and Stochastic Gradient Descent (SGD). These models were trained on the preprocessed dataset, and their respective predictions were compared.
Plotted a line to show how the model could have behaved using PCA.




Plotted a simple curve to show the difference we are getting from ‘actual y’ and ‘y predicted’.



**Task 4 (Evaluation)**

Model evaluation metrics, including R-squared (R2) score, Mean Squared Error (MSE), and Mean Absolute Error (MAE), were computed for each regression model. The performance of the models was visualized through bar plots, providing a clear comparison of their accuracy and predictive power.

| NAME | R2_SCORE | MSE | MAE |
|---|---|---|---|
| Batch | -3.9567 | 0.179493 | 0.358550 |
| MiniBatch | 0.6282 | 0.013462 | 0.085906 |
| Stochastic | 0.6555 | 0.014362 | 0.085906 |

From the evaluation, it was observed that the Stochastic Gradient Descent (SGD) algorithm performed the best on this dataset, achieving the highest R2 score and demonstrating superior accuracy in predicting housing prices.

The comparison plot and the tabular representation in the report facilitate a quick overview of the model performances and assist in selecting the most suitable algorithm for the given regression problem.