BRAC UNIVERSITY

Inspiring Excellence

# ***Project Report***

## CSE422 : Artificial Intelligence

## **Project Title** : *Movie review System*

| Group No : 07 , CSE-422 Lab Section : 12, Spring 2025 | |
|---|---|
| **ID** | **Name** |
| 23341100 | Waseque Chowdhury |

**Table of Contents**:

# 1. Introduction

In the modern film industry, data-driven insights are increasingly vital for making informed decisions about movie production, marketing, and audience targeting. This project aims to build a machine learning model that can classify movies into different rating categories (e.g., Excellent, Good, Average, Poor) based on various features such as budget, genre, director, actor popularity, awards, and promotional investment. By analyzing historical movie data, the goal is to predict a movie's likely reception category prior to its release.

This project aims to empower producers and stakeholders with predictive tools that can estimate audience response, optimize investment strategies, and enhance decision-making in early production stages. Using classification algorithms and neural networks, the project explores relationships between movie attributes and their rating outcomes, ultimately contributing to smarter entertainment analytics.

# 2. Dataset Description

The dataset used in this project contains a variety of features related to movies, including both numerical and categorical attributes. These features encompass information such as the movie’s budget, runtime, release year, genre, director, actor popularity, number of awards won, and marketing spend. The target variable is Rating_Category, which classifies each movie into a predefined rating group such as Excellent, Good, Average, or Poor.

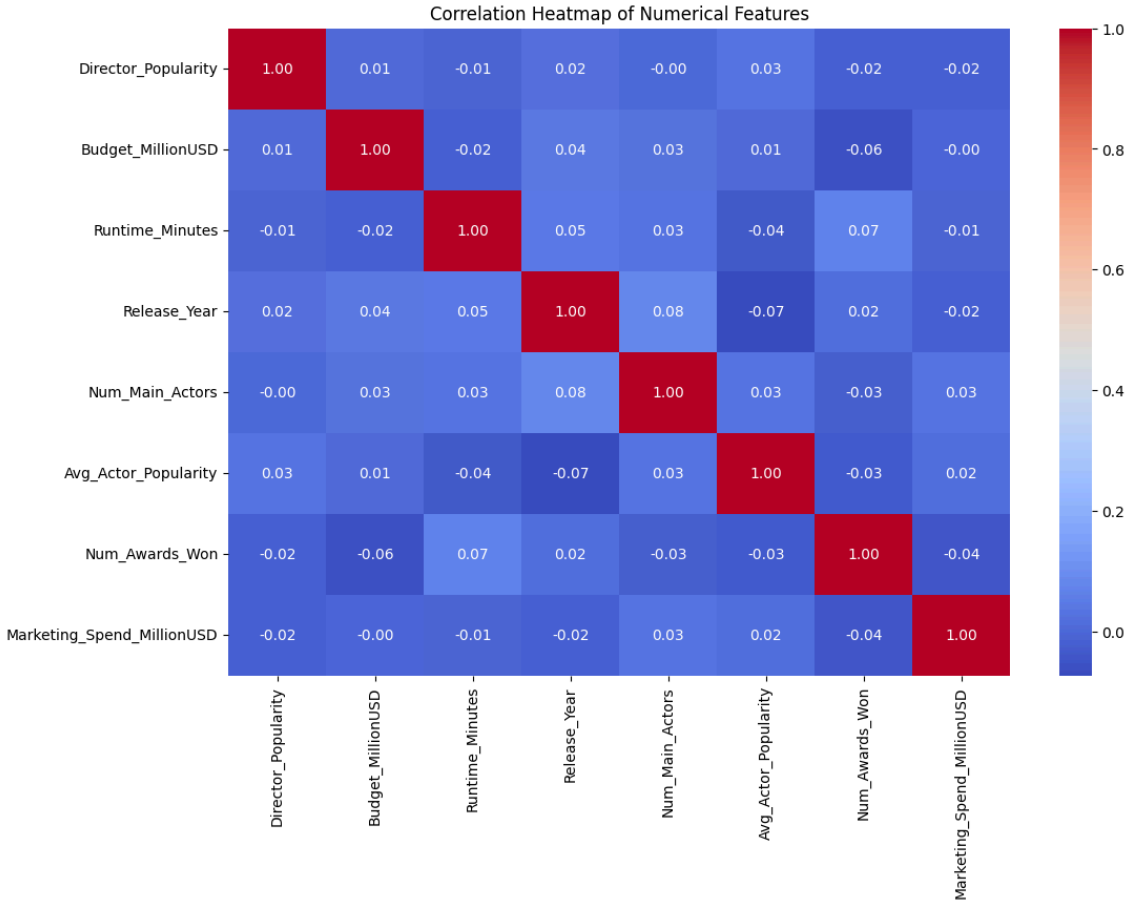
- Data points: 1200
- Features: 12
- Prediction task: Classification. The target variable, "Rating_Category," is categorical, representing different rating levels.

- Feature Types:
    - **Quantitative:** Director_Popularity, Budget_MillionUSD, Runtime_Minutes, Release_Year, Num_Main_Actors, Avg_Actor_Popularity, Num_Awards_Won, Marketing_Spend_MillionUSD
    - **Categorical:** Genre, Has_Famous_Producer, Is_Sequel, Rating_Category

**Libraries Used:** The project uses the following libraries:

- Pandas- for data manipulation and analysis
- Numpy for numerical operations
- Sklearn- for machine learning models and preprocessing
- Matplotlib- for data visualization
- Seaborn- for enhanced visualization

## *Correlation Heatmap*

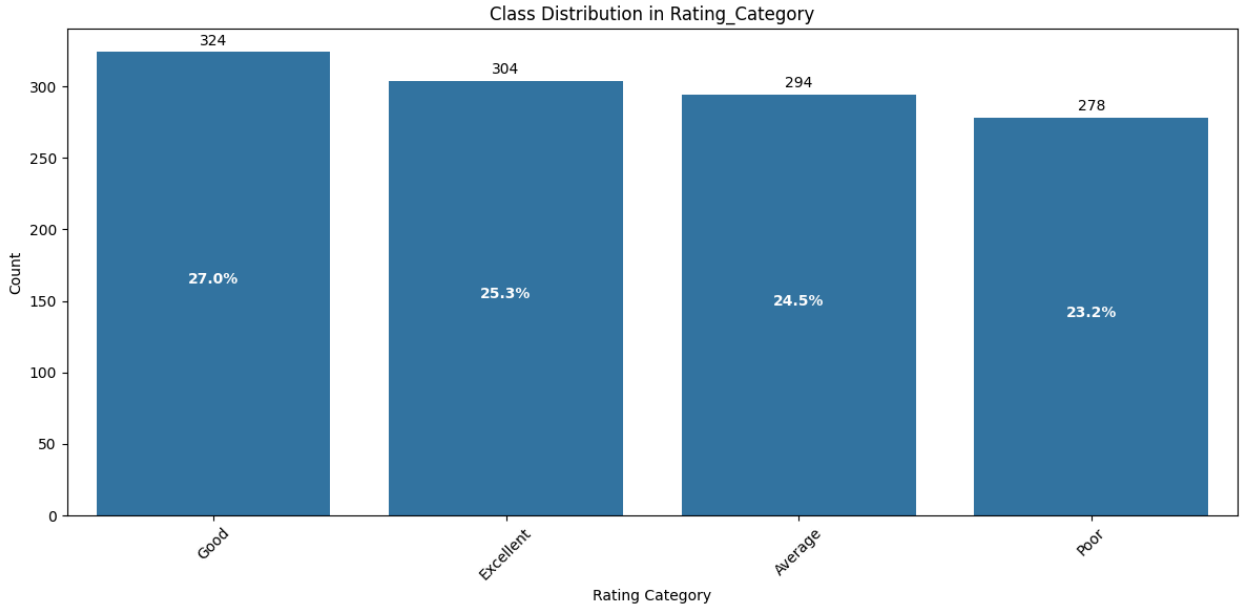


The heatmap shows strong correlations between Budget and Marketing Spend features, indicating they increase together. These insights can guide feature selection and interpretation.

## Imbalanced Dataset

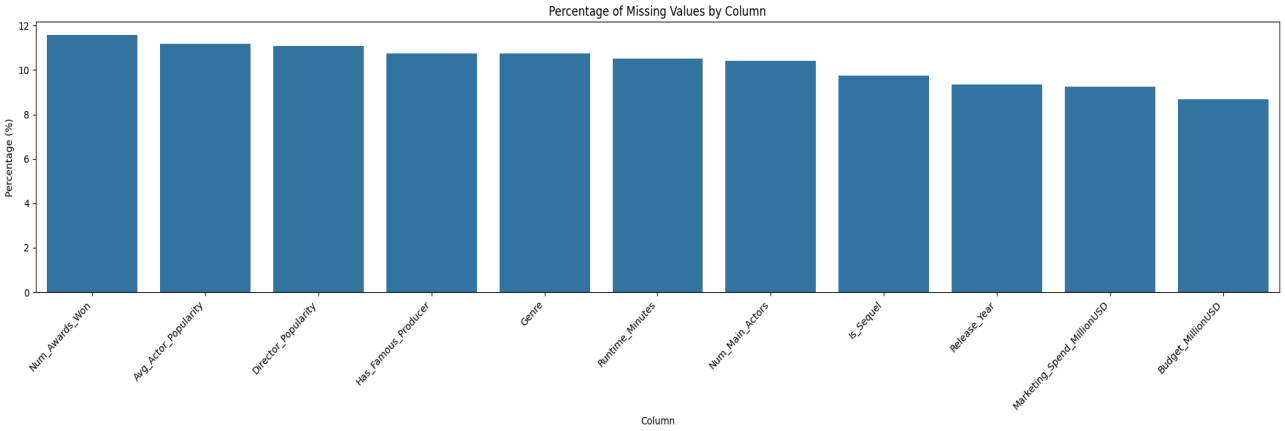
The dataset exhibits a relatively balanced class distribution for the target variable "Rating_Category," with no single class dominating significantly.

- Good: 27%
- Excellent: 25.3%
- Average: 24.5%
- Poor: 23.2%

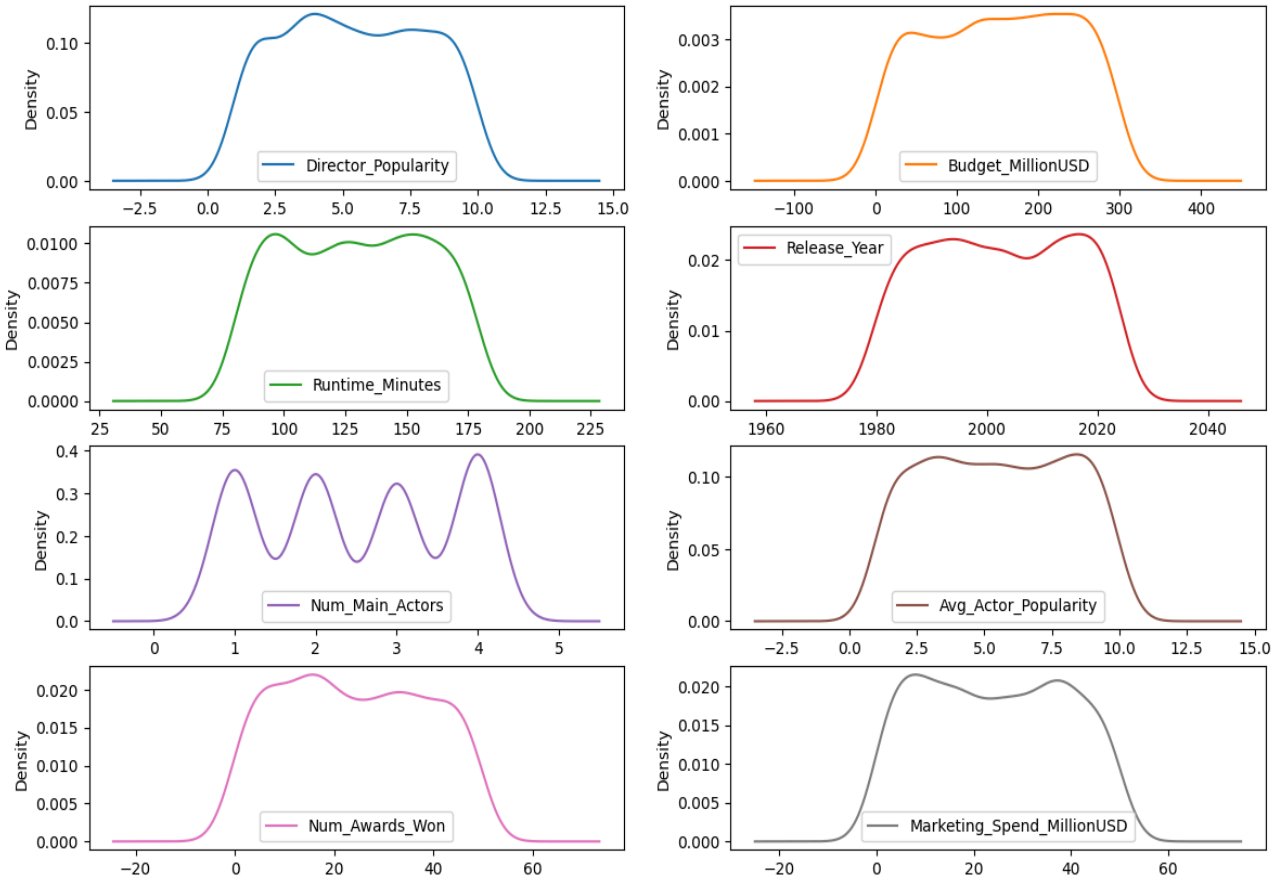


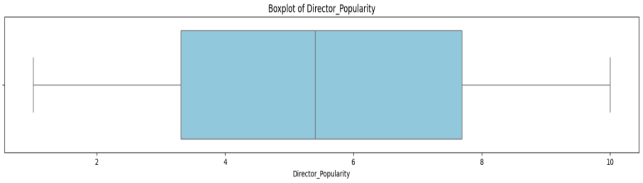

### *Exploratory Data Analysis*

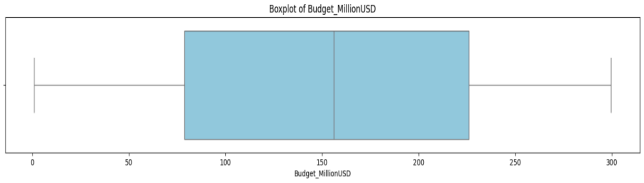
EDA revealed patterns and relationships within the data. For example, movies with higher budgets and marketing spends tended to receive better ratings. Box plots and histograms further illustrated the distribution of numerical features across different rating categories.

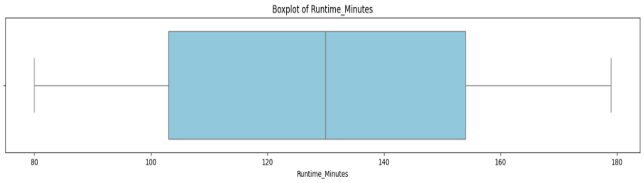


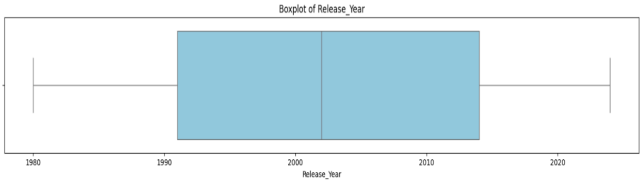

Density plot of Numerical features

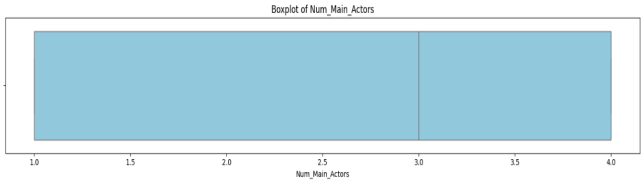


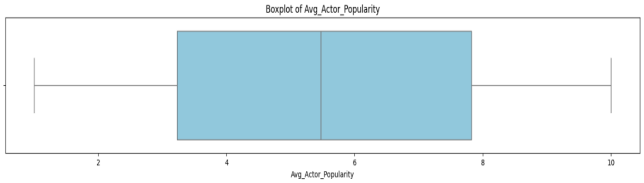

Boxplot of Director_Popularity
2
4
6
8
10
Director_Popularity

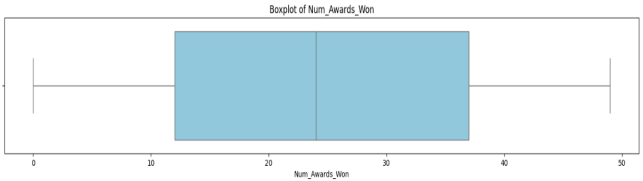

Boxplot of Budget_MillionUSD
0
50
100
150
200
250
300
Budget_MillionUSD

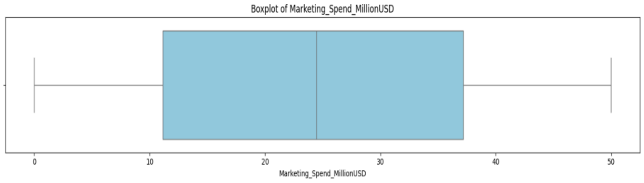

Boxplot of Runtime_Minutes
80
100
120
140
160
180
Runtime_Minutes

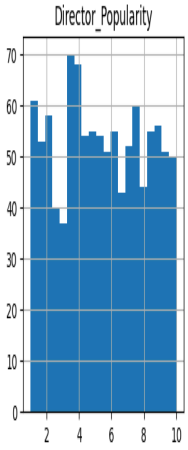

Boxplot of Release_Year
1980
1990
2000
2010
2020
Release_Year

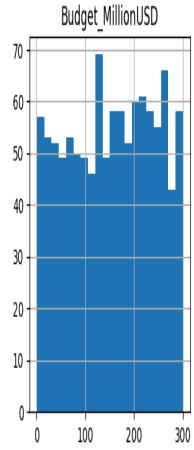

Boxplot of Num_Main_Actors
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Num_Main_Actors

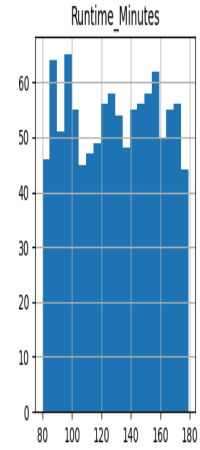

Boxplot of Avg_Actor_Popularity
2
4
6
8
10
Avg_Actor_Popularity

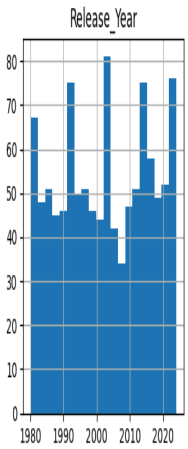

Boxplot of Num_Awards_Won
0
10
20
30
40
50
Num_Awards_Won

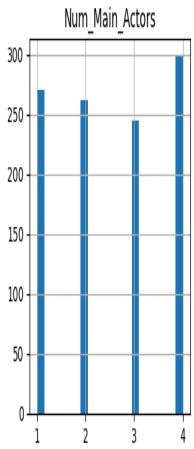

Boxplot of Marketing_Spend_MillionUSD
0
10
20
30
40
50
Marketing_Spend_MillionUSD

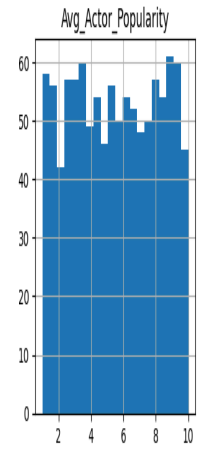

Director_Popularity
0
10
20
30
40
50
60
70
2
4
6
8
10

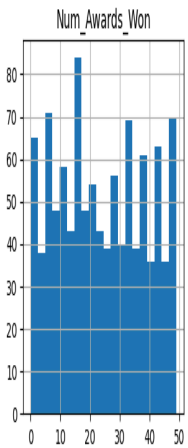

Budget_MillionUSD
0
10
20
30
40
50
60
70
0
100
200
300

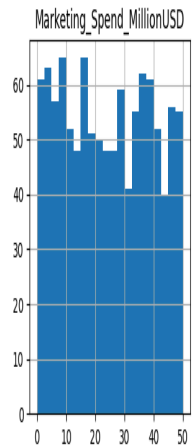

Runtime_Minutes
0
10
20
30
40
50
60
80
100
120
140
160
180


Release_Year
0
10
20
30
40
50
60
70
80
1980
1990
2000
2010
2020


Num_Main_Actors
0
50
100
150
200
250
300
1
2
3
4


Avg_Actor_Popularity
0
10
20
30
40
50
60
2
4
6
8
10


Num_Awards_Won
0
10
20
30
40
50
60
70
80
0
10
20
30
40
50


Marketing_Spend_MillionUSD
0
10
20
30
40
50
60
0
10
20
30
40
50

## 3) Dataset Pre-processing:

**Problem 1:** Missing / Null Values

The dataset contains missing values in both numerical and categorical columns.

**Solution:**

- **Numerical Columns**: Imputed using the **median** value.
  *Cause*: Median is robust to outliers and ensures that skewed distributions don't affect imputation.

- **Categorical Columns**: Imputed using the **most frequent (mode)** value.
  *Cause*: This helps preserve the original distribution and avoids introducing unknown categories.

**Problem 2:** Categorical Variables

The dataset contains categorical features like:

- Genre
- Has_Famous_Producer
- Is_Sequel

**Solution:** Applied **One-Hot Encoding** using OneHotEncoder.
Cause: One-hot encoding avoids ordinal misinterpretation and is compatible with most machine learning models.

**Problem 3:** Feature Scaling and Pipeline

Numerical features like Budget_MillionUSD, Runtime_Minutes, etc., had varied scales, which could negatively impact model training.

**Solution:** Applied **Min-Max Normalization** using MinMaxScaler to scale all numerical values between 0 and 1. By standardizing the preprocessing steps and model evaluation, we created a robust framework for comparing different algorithms. This methodology not only improves the reliability of our results but also establishes a foundation for future model iterations and deployment.

*Cause*: Brings all features to the same range, improving model convergence and distance-based algorithm performance (like KNN).

## 4) Dataset Splitting:

Since our target variable, Rating_Category, is categorical and exhibits class imbalance, **stratified splitting** is used to ensure the same proportion of classes in both training and testing sets.

- **Train set**: 70% of the data
- **Test set**: 30% of the data
- **Splitting Method**: StratifiedShuffleSplit from sklearn.model_selection (or train_test_split(..., stratify=y))

## ❖ Data Handling

- Missing Values: Missing values in numerical features were imputed using the median, and missing values in categorical features were imputed using the most frequent category.
- Categorical Features: Categorical features (Genre, Has_Famous_Producer, Is_Sequel) were one-hot encoded.
- Scaling: Numerical features were scaled to the range [0, 1] using MinMaxScaler.
- Pipeline: Most-frequent imputation for both numerical and categorical features

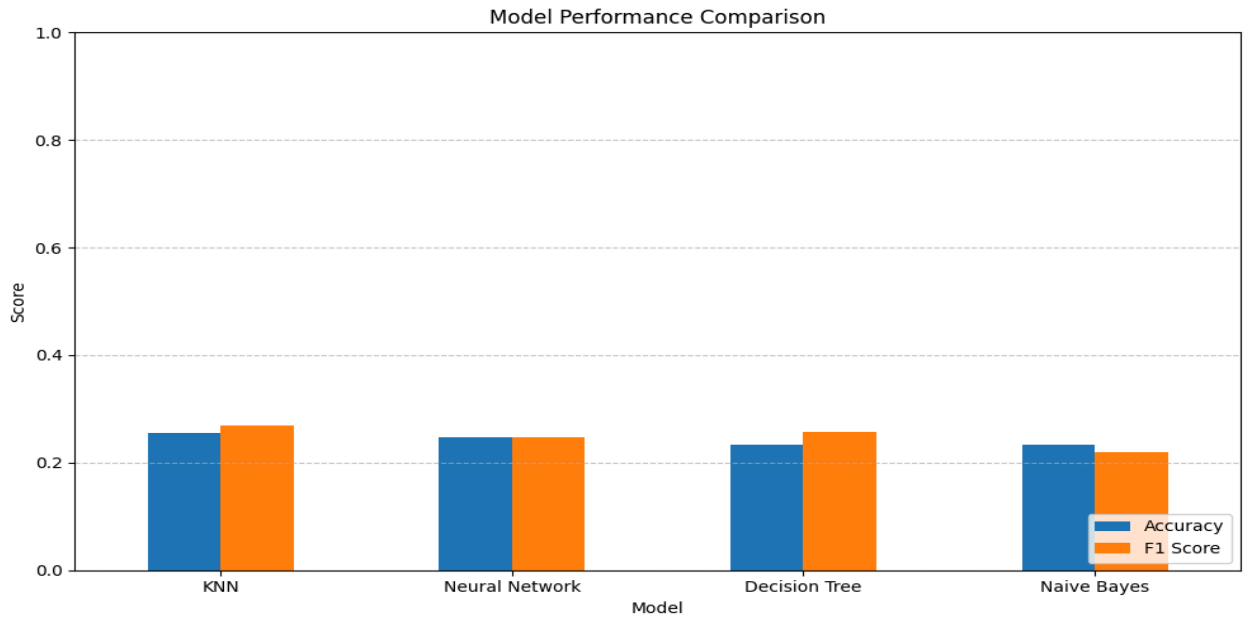
## 5) Model Training & Testing:

Five classification models were trained and tested:

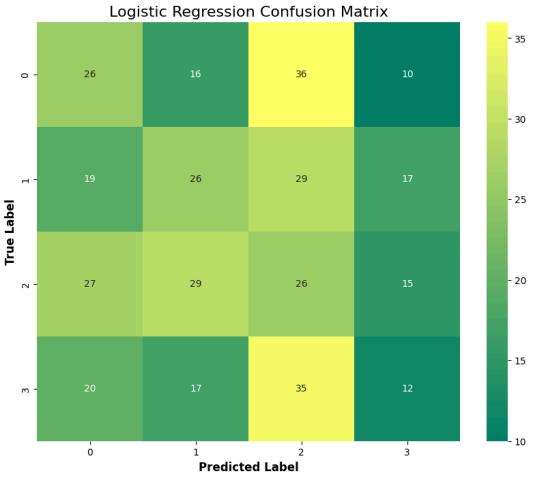
- Neural Network: MLPClassifier(hidden_layer_sizes=(128,64, 32), activation='relu', learning rate = ‘0.001’, solver='adam', max_iter=1000, random_state=42)
- KNN: KNeighborsClassifier(n_neighbors=5)
- Decision Tree: DecisionTreeClassifier(max_depth=5)
- Logistic Regression: LogisticRegression(C=1.0, max_iter=1000)
- Naive Bayes: GaussianNB()

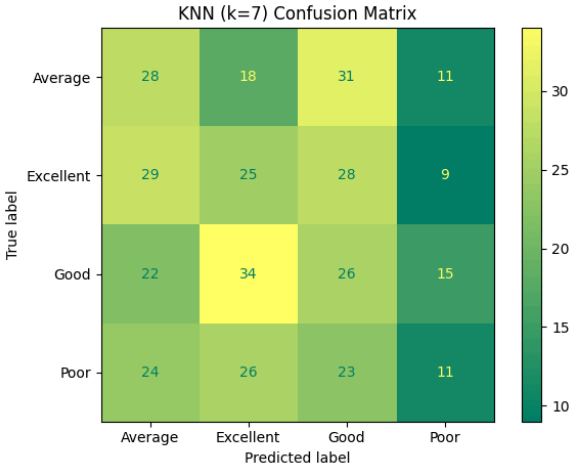
## 6) Model Comparison:

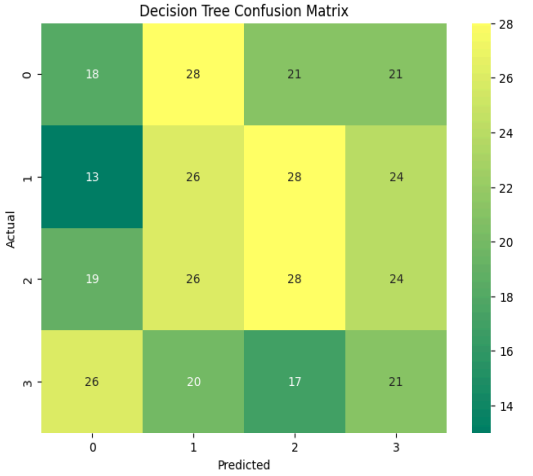
**Bar chart showcasing the prediction accuracy of all models:**

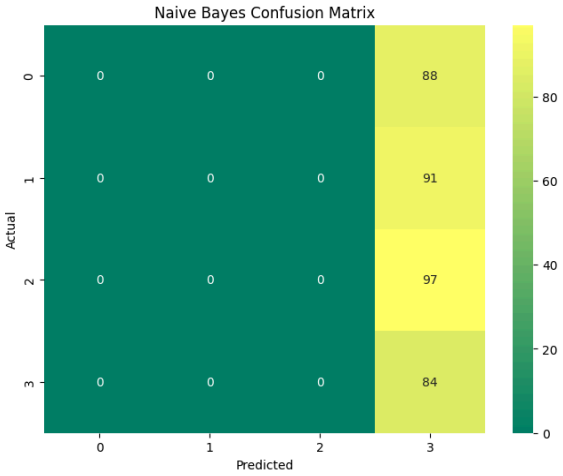


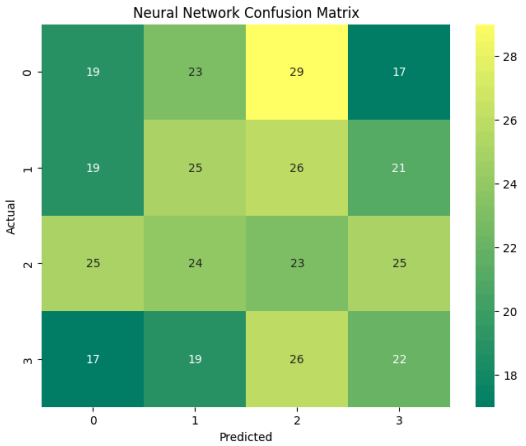
# Confusion matrix



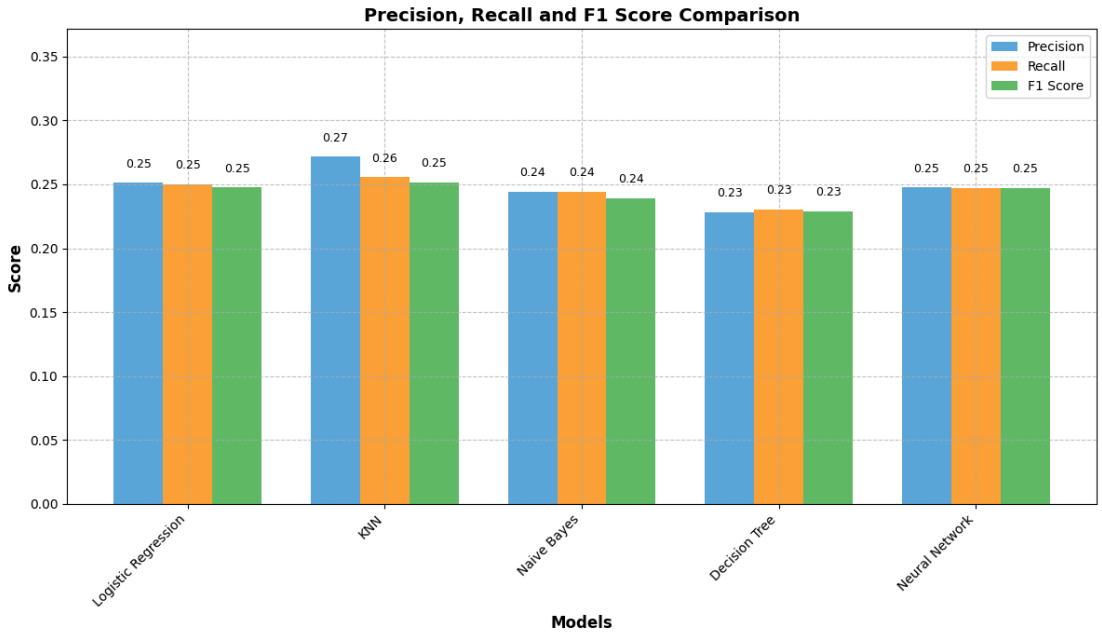



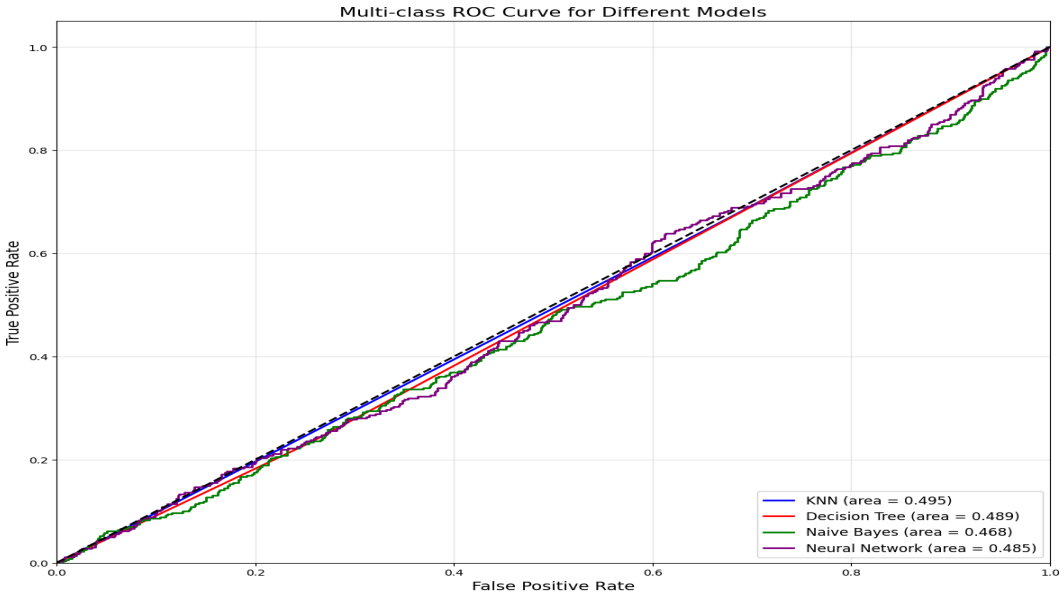



## Precision and recall comparison of each model:




## AUC Score and ROC Curve:




All models demonstrated poor performance with low accuracy, precision, and recall scores. The neural network failed to converge even with an increased number of iterations.

## 6) Conclusion:

While the project aimed to predict movie ratings, the trained models performed poorly. This indicates potential challenges with the training process or limitations in the dataset. Further investigation is needed to improve model performance. Exploring different model architectures, hyperparameters, feature engineering techniques, and addressing convergence issues are crucial next steps.

### Challenges Faced

- Data type mismatches during model evaluation were encountered and resolved.
- Zero division issues during precision calculation were handled using the zero_division parameter in precision_score.
- All models showed poor performance.
- The neural network did not converge.

## 7) Reference:


[1] NumPy Contributors, "NumPy," NumPy, 2025. [Online]. Available: https://numpy.org/

[2] Pandas Contributors, "pandas - Python Data Analysis Library," 2025. [Online]. Available: https://pandas.pydata.org/

[3] The Matplotlib Development Team, "Matplotlib: Visualization with Python," 2025. [Online]. Available: https://matplotlib.org/

[4] The Seaborn Development Team, "Seaborn: statistical data visualization," 2025. [Online]. Available: https://seaborn.pydata.org/

[5] The scikit-learn developers, "scikit-learn: Machine Learning in Python," 2025. [Online]. Available: https://scikit-learn.org/

[6] Martín Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2025. [Online]. Available: https://www.tensorflow.org/

[7] The Keras Team, "Keras," 2025. [Online]. Available: https://keras.io/

[8] Dataset : Movie Rating Dataset (link)