ML CAC-2

Name: Kingsuk Rakshit

Reg No: 23122121

Introduction

In this project, we aim to explore and analyze a dataset containing information about various movies released in the year 1980. We will employ various machine learning techniques to understand the data and build predictive models. The project encompasses data preprocessing, exploratory data analysis (EDA), and the application of machine learning algorithms to derive meaningful insights from the dataset.

Dataset Description

The dataset used in this project provides comprehensive details about movies released in 1980. Each entry in the dataset represents a single movie, described by various attributes:

1. **name**: *string* - The title of the movie.
2. **rating**: *string* - The MPAA rating of the movie (e.g., R, PG).
3. **genre**: *string* - The genre of the movie.
4. **year released**: *string* - The year the movie was released.
5. **score**: *float* - The IMDb user rating score for the movie.
6. **votes**: *integer* - The number of IMDb user votes for the movie.
7. **director**: *string* - The name of the movie's director.
8. **writer**: *string* - The name of the movie's writer.
9. **star**: *string* - The name of the movie's main star.
10. **country**: *string* - The country where the movie was produced.
11. **budget**: *integer* - The budget of the movie in US dollars.
12. **gross**: *integer* - The gross revenue of the movie in US dollars.
13. **company**: *string* - The production company behind the movie.
14. **runtime**: *integer* - The length of the movie in minutes.

The dataset serves as a rich resource for analyzing various aspects of movie production and reception, including financial metrics (budget, gross revenue), viewer metrics (score, votes), and categorical attributes (genre, rating, country).

Import Libraries and Load Dataset

In [19]:
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split

# Load the dataset
data = pd.read_csv('movies.csv')

# Display the first few rows of the dataset
data.head()
```

Out[19]:

	name	rating	genre	year	released	score	votes	director	write
0	The Shining	R	Drama	1980	June 13, 1980 (United States)	8.4	927000.0	Stanley Kubrick	Stepher King
1	The Blue Lagoon	R	Adventure	1980	July 2, 1980 (United States)	5.8	65000.0	Randal Kleiser	Henry De Vere Stacpoole
2	Star Wars: Episode V - The Empire Strikes Back	PG	Action	1980	June 20, 1980 (United States)	8.7	1200000.0	Irvin Kershner	Leigh Bracket
3	Airplane!	PG	Comedy	1980	July 2, 1980 (United States)	7.7	221000.0	Jim Abrahams	Jim Abraham
4	Caddyshack	R	Comedy	1980	July 25, 1980 (United States)	7.3	108000.0	Harold Ramis	Brian Doyle Murray

In [20]:
```
# Display last few rows
data.tail()
```

Out[20]:

	name	rating	genre	year	released	score	votes	director	writer	
7663	More to Life	NaN	Drama	2020	October 23, 2020 (United States)	3.1	18.0	Joseph Ebanks	Joseph Ebanks	Sha
7664	Dream Round	NaN	Comedy	2020	February 7, 2020 (United States)	4.7	36.0	Dusty Dukatz	Lisa Huston	M Sa
7665	Saving Mbango	NaN	Drama	2020	April 27, 2020 (Cameroon)	5.7	29.0	Nkanya Nkwai	Lynno Lovert	Or
7666	It's Just Us	NaN	Drama	2020	October 1, 2020 (United States)	NaN	NaN	James Randall	James Randall	Ch
7667	Tee em el	NaN	Horror	2020	August 19, 2020 (United States)	5.7	7.0	Pereko Mosia	Pereko Mosia	Siyab M

Handle Missing Values and Encode Categorical Data

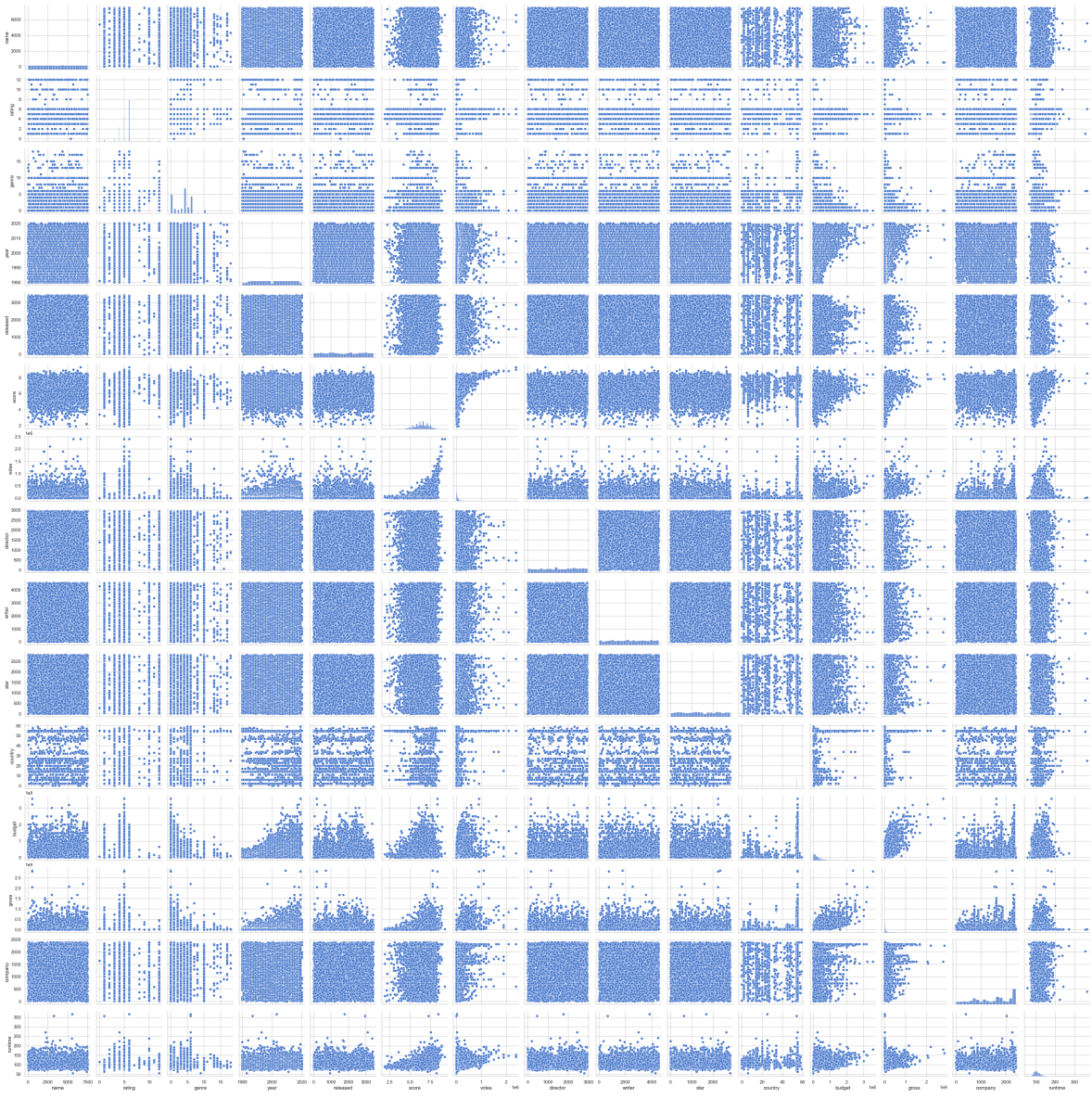
In [21]:
```
# Handle missing values
data.fillna(data.mean(numeric_only=True), inplace=True)
```

In [22]:
```
# Convert categorical features to numerical using Label Encoding
from sklearn.preprocessing import LabelEncoder

# Initialize label encoder object
label_encoders = {}

# Iterate through categorical columns and encode them
for column in data.select_dtypes(include=['object']).columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])
```

Exploratory Data Analysis (EDA)

Pairplot

In [38]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Pairplot to visualize relationships between features
sns.pairplot(data)
plt.show()
```

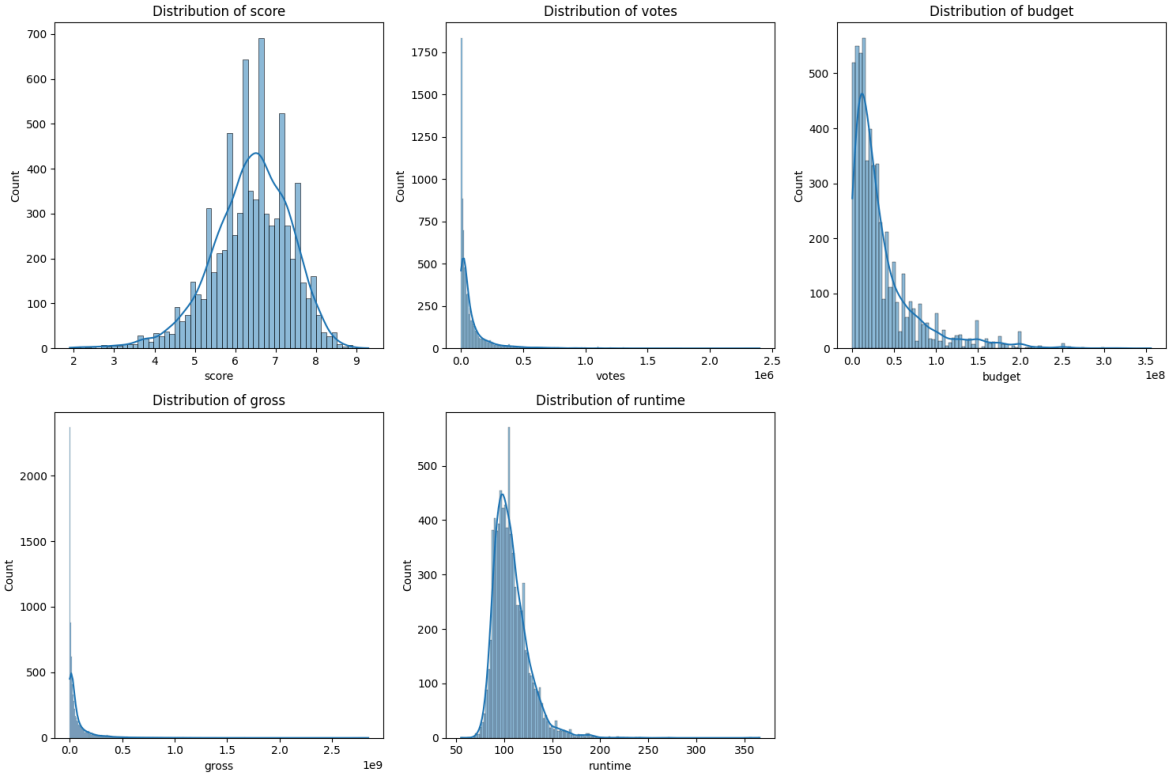


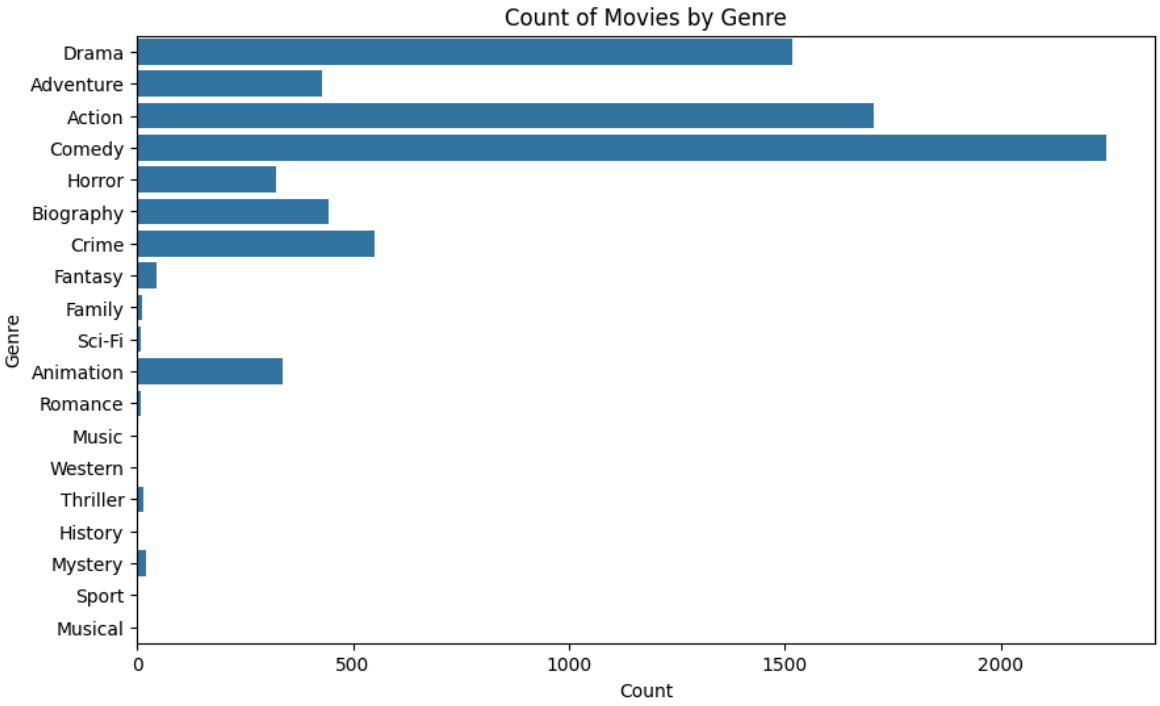
In [24]:
```
# Import necessary libraries for EDA
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
data = pd.read_csv('movies.csv')

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(data.head())

# Check summary statistics of numerical columns
print("Summary statistics of numerical columns:")
print(data.describe())

# Check the data types and missing values
print("Data types and missing values:")
print(data.info())

# Visualize the distribution of numerical features
numerical_features = ['score', 'votes', 'budget', 'gross', 'runtime']
plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features, 1):
```

```
    plt.subplot(2, 3, i)
    sns.histplot(data[feature], kde=True)
    plt.title(f"Distribution of {feature}")
plt.tight_layout()
plt.show()

# Visualize the count of movies by genre
plt.figure(figsize=(10, 6))
sns.countplot(y='genre', data=data)
plt.title("Count of Movies by Genre")
plt.xlabel("Count")
plt.ylabel("Genre")
plt.show()
```

```
First few rows of the dataset:
                                        name rating      genre  year  \
0                                The Shining      R      Drama  1980
1                              The Blue Lagoon      R  Adventure  1980
2  Star Wars: Episode V - The Empire Strikes Back     PG     Action  1980
3                                   Airplane!     PG     Comedy  1980
4                                  Caddyshack      R     Comedy  1980

                      released  score      votes         director  \
0  June 13, 1980 (United States)    8.4   927000.0  Stanley Kubrick
1   July 2, 1980 (United States)    5.8    65000.0   Randal Kleiser
2  June 20, 1980 (United States)    8.7  1200000.0   Irvin Kershner
3   July 2, 1980 (United States)    7.7   221000.0     Jim Abrahams
4  July 25, 1980 (United States)    7.3   108000.0     Harold Ramis

                   writer            star         country      budget  \
0            Stephen King  Jack Nicholson  United Kingdom  19000000.0
1  Henry De Vere Stacpoole  Brooke Shields   United States   4500000.0
2          Leigh Brackett     Mark Hamill   United States  18000000.0
3            Jim Abrahams     Robert Hays   United States   3500000.0
4       Brian Doyle-Murray     Chevy Chase   United States   6000000.0

         gross             company  runtime
0   46998772.0        Warner Bros.    146.0
1   58853106.0   Columbia Pictures    104.0
2  538375067.0           Lucasfilm    124.0
3   83453539.0  Paramount Pictures     88.0
4   39846344.0      Orion Pictures     98.0
Summary statistics of numerical columns:
              year        score         votes        budget         gross  \
count  7668.000000  7665.000000  7.665000e+03  5.497000e+03  7.479000e+03
mean   2000.405451     6.390411  8.810850e+04  3.558988e+07  7.850054e+07
std      11.153508     0.968842  1.633238e+05  4.145730e+07  1.657251e+08
min    1980.000000     1.900000  7.000000e+00  3.000000e+03  3.090000e+02
25%    1991.000000     5.800000  9.100000e+03  1.000000e+07  4.532056e+06
50%    2000.000000     6.500000  3.300000e+04  2.050000e+07  2.020576e+07
75%    2010.000000     7.100000  9.300000e+04  4.500000e+07  7.601669e+07
max    2020.000000     9.300000  2.400000e+06  3.560000e+08  2.847246e+09

           runtime
count  7664.000000
mean    107.261613
std      18.581247
min      55.000000
25%      95.000000
50%     104.000000
75%     116.000000
max     366.000000
Data types and missing values:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7668 entries, 0 to 7667
Data columns (total 15 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      7668 non-null   object
 1   rating    7591 non-null   object
 2   genre     7668 non-null   object
 3   year      7668 non-null   int64
 4   released  7666 non-null   object
 5   score     7665 non-null   float64
```

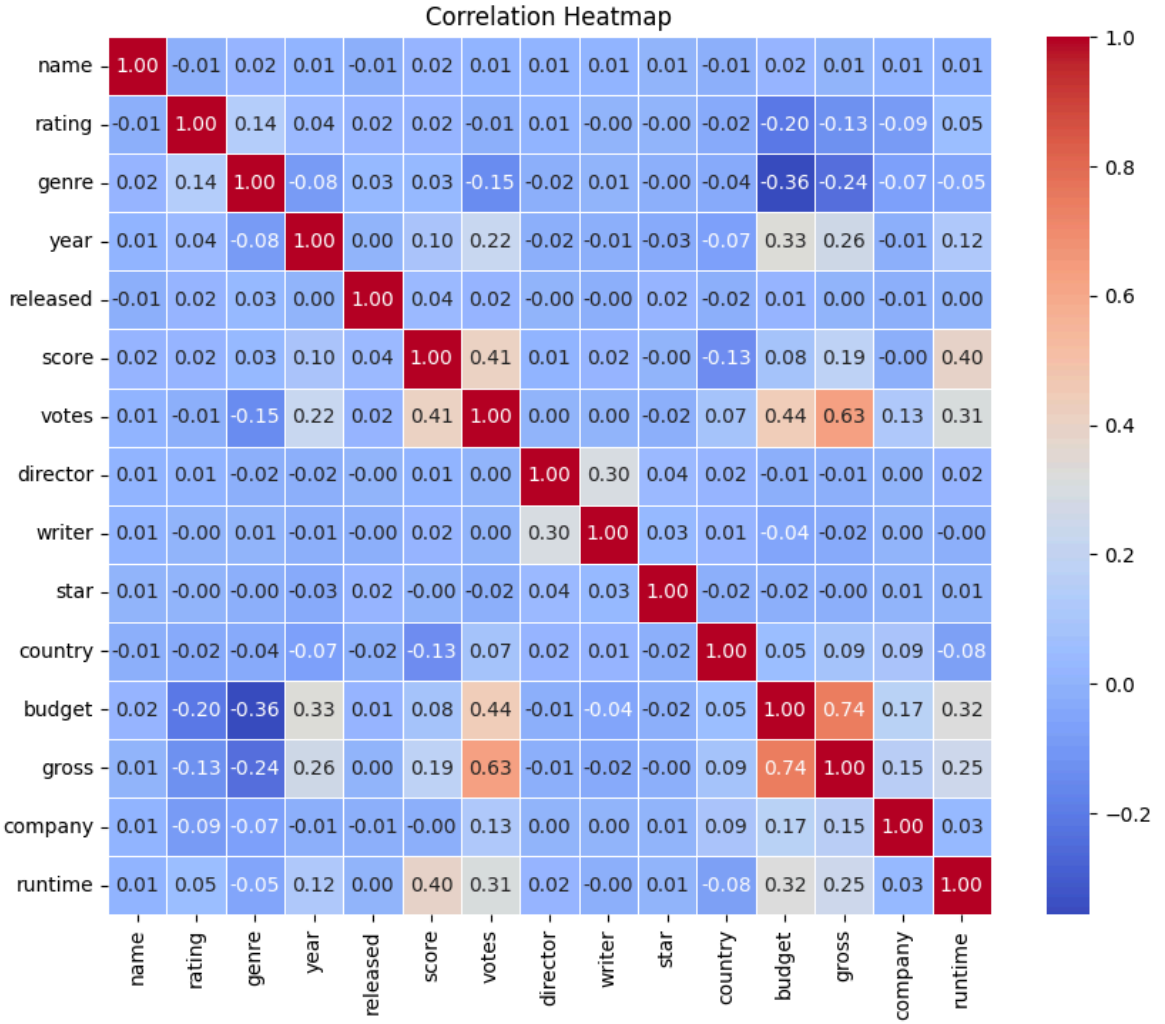
```
 6   votes     7665 non-null   float64
 7   director  7668 non-null   object
 8   writer    7665 non-null   object
 9   star      7667 non-null   object
 10  country   7665 non-null   object
 11  budget    5497 non-null   float64
 12  gross     7479 non-null   float64
 13  company   7651 non-null   object
 14  runtime   7664 non-null   float64
dtypes: float64(5), int64(1), object(9)
memory usage: 898.7+ KB
None
```





Correlation Heatmap

In [31]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Compute correlation matrix
corr = data.corr()

# Generate a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



Define Features and Target Variable

In [25]:
```
# Encode categorical features
data_encoded = pd.get_dummies(data, columns=['rating', 'director', 'writer', 'st

# Separate features and target variable
X = data_encoded.drop(columns=['genre', 'name', 'year'])
y = data_encoded['genre']

# Split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

Train and Evaluate Multiple Models

In [26]:
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the dataset
data = pd.read_csv('movies.csv')

# Handle missing values
data.fillna(data.mean(numeric_only=True), inplace=True)

# Convert categorical features to numerical using Label Encoding
label_encoders = {}
for column in data.select_dtypes(include=['object']).columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])

# Separate features and target variable
X = data.drop(columns=['genre'])
y = data['genre']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Check for missing values in the training set by column
missing_values = X_train.columns[X_train.isnull().any()]
if missing_values.any():
    print("Columns with missing values in the training set:")
    for col in missing_values:
        print(f"{col}: {X_train[col].isnull().sum()} missing values")
else:
    print("No missing values found in the training set.")

# Impute missing values with the mean of the column
X_train['score'].fillna(X_train['score'].mean(), inplace=True)

# List of classifiers to evaluate
classifiers = [
    KNeighborsClassifier(n_neighbors=5),
    LogisticRegression(random_state=42),
    GaussianNB(),
    MLPClassifier(random_state=42),
    RandomForestClassifier(random_state=42),
    GradientBoostingClassifier(random_state=42)
]

# Function to evaluate classifiers
def evaluate_classifier(model, X_test, y_test):
    y_pred = model.predict(X_test)
    print(f"{model.__class__.__name__} Classification Report:")
```

```
    print(classification_report(y_test, y_pred))
    print(f"{model.__class__.__name__} Accuracy:", accuracy_score(y_test, y_pred
    print("="*60)

# Train and evaluate each classifier
for model in classifiers:
    model.fit(X_train, y_train)
    evaluate_classifier(model, X_test, y_test)
```

```
No missing values found in the training set.
KNeighborsClassifier Classification Report:
              precision    recall  f1-score   support

           0       0.32      0.50      0.39       336
           1       0.11      0.04      0.06        95
           2       0.14      0.06      0.08        68
           3       0.08      0.06      0.06        90
           4       0.36      0.49      0.41       443
           5       0.06      0.02      0.03       110
           6       0.27      0.21      0.24       304
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00        13
          10       0.23      0.08      0.12        59
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          17       0.00      0.00      0.00         4

    accuracy                           0.30      1534
   macro avg       0.10      0.10      0.09      1534
weighted avg       0.26      0.30      0.27      1534

KNeighborsClassifier Accuracy: 0.3044328552803129
============================================================
```

```
C:\Users\raksh\AppData\Local\Temp\ipykernel_19264\910759086.py:41: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained as
signment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work becau
se the intermediate object on which we are setting values always behaves as a cop
y.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.meth
od({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to pe
rform the operation inplace on the original object.


  X_train['score'].fillna(X_train['score'].mean(), inplace=True)
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\linear_model\_l
ogistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
```

```
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
LogisticRegression Classification Report:
              precision    recall  f1-score   support

           0       0.27      0.69      0.39       336
           1       0.00      0.00      0.00        95
           2       0.06      0.06      0.06        68
           3       0.00      0.00      0.00        90
           4       0.31      0.43      0.36       443
           5       0.00      0.00      0.00       110
           6       0.00      0.00      0.00       304
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00        13
          10       0.00      0.00      0.00        59
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          17       0.00      0.00      0.00         4

    accuracy                           0.28      1534
   macro avg       0.04      0.08      0.05      1534
weighted avg       0.15      0.28      0.19      1534

LogisticRegression Accuracy: 0.2770534550195567
============================================================
GaussianNB Classification Report:
              precision    recall  f1-score   support

           0       0.43      0.26      0.33       336
           1       0.00      0.00      0.00        95
           2       0.25      0.03      0.05        68
           3       0.00      0.00      0.00        90
           4       0.32      0.90      0.47       443
           5       0.08      0.01      0.02       110
           6       0.29      0.03      0.06       304
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00        13
          10       0.00      0.00      0.00        59
          11       0.00      0.00      0.00         0
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          16       0.00      0.00      0.00         0
          17       0.00      0.00      0.00         4

    accuracy                           0.33      1534
   macro avg       0.08      0.07      0.05      1534
weighted avg       0.26      0.33      0.22      1534

GaussianNB Accuracy: 0.3259452411994785
============================================================
MLPClassifier Classification Report:
              precision    recall  f1-score   support

           0       0.30      0.24      0.27       336
           1       0.07      0.02      0.03        95
           2       0.10      0.56      0.17        68
           3       0.00      0.00      0.00        90
           4       0.43      0.09      0.14       443
```

```
           5       0.09      0.13      0.11       110
           6       0.27      0.47      0.34       304
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00        13
          10       0.00      0.00      0.00        59
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          17       0.00      0.00      0.00         4

    accuracy                           0.21      1534
   macro avg       0.08      0.10      0.07      1534
weighted avg       0.26      0.21      0.18      1534

MLPClassifier Accuracy: 0.20599739243807041
============================================================
```

```
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
RandomForestClassifier Classification Report:
              precision    recall  f1-score   support

           0       0.51      0.50      0.51       336
           1       0.56      0.05      0.10        95
           2       0.76      0.71      0.73        68
           3       0.27      0.08      0.12        90
           4       0.42      0.67      0.52       443
           5       0.57      0.04      0.07       110
           6       0.36      0.44      0.40       304
           7       0.00      0.00      0.00         1
           8       0.00      0.00      0.00        13
          10       0.25      0.07      0.11        59
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          17       0.00      0.00      0.00         4

    accuracy                           0.44      1534
   macro avg       0.25      0.17      0.17      1534
weighted avg       0.44      0.44      0.39      1534

RandomForestClassifier Accuracy: 0.43546284224250326
============================================================
```

```
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
GradientBoostingClassifier Classification Report:
              precision    recall  f1-score   support

           0       0.53      0.49      0.51       336
           1       0.39      0.12      0.18        95
           2       0.74      0.75      0.74        68
           3       0.24      0.12      0.16        90
           4       0.45      0.69      0.55       443
           5       0.21      0.05      0.09       110
           6       0.42      0.45      0.43       304
           7       0.00      0.00      0.00         1
           8       0.25      0.08      0.12        13
          10       0.36      0.22      0.27        59
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         5
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         3
          16       0.00      0.00      0.00         0
          17       0.00      0.00      0.00         4
          18       0.00      0.00      0.00         0

    accuracy                           0.46      1534
   macro avg       0.21      0.17      0.18      1534
weighted avg       0.43      0.46      0.43      1534

GradientBoostingClassifier Accuracy: 0.45632333767926986
============================================================
```

```
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

Decision Tree

In [27]:

```python
# Import necessary libraries
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Load the dataset
data = pd.read_csv('movies.csv')

# Preprocessing: Convert categorical variables to numerical
label_encoder = LabelEncoder()
for column in data.columns:
    if data[column].dtype == 'object':
        data[column] = label_encoder.fit_transform(data[column])

# Split the data into features (X) and target (y)
X = data.drop('rating', axis=1)  # Drop the target column
y = data['rating']
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Create and train the Decision Tree classifier
dt_classifier = DecisionTreeClassifier(random_state=42)
dt_classifier.fit(X_train, y_train)

# Evaluate the Decision Tree classifier
y_pred_dt = dt_classifier.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print("Accuracy of Decision Tree Classifier:", accuracy_dt)
print("Classification Report of Decision Tree Classifier:")
print(classification_report(y_test, y_pred_dt))
```

```
Accuracy of Decision Tree Classifier: 0.4765319426336376
Classification Report of Decision Tree Classifier:
              precision    recall  f1-score   support

           1       0.29      0.29      0.29        35
           2       0.00      0.00      0.00         6
           3       0.29      0.29      0.29        59
           4       0.34      0.33      0.34       271
           5       0.44      0.42      0.43       422
           6       0.58      0.61      0.60       710
           8       0.00      0.00      0.00         3
           9       0.00      0.00      0.00         0
          10       0.08      0.08      0.08        12
          11       0.00      0.00      0.00         1
          12       0.00      0.00      0.00        15

    accuracy                           0.48      1534
   macro avg       0.18      0.18      0.18      1534
weighted avg       0.47      0.48      0.47      1534
```

```
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Recall is ill-defined and being set to
0.0 in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

SVM

In [28]:

```
# Import SVM classifier
from sklearn.svm import SVC

# Create and train the SVM model
```

```
svm = SVC(kernel='linear', random_state=42)
svm
```

Out[28]:

SVC

SVC(kernel='linear', random_state=42)

In [29]:

```
# Import necessary libraries
import pandas as pd
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer

# Load the dataset
data = pd.read_csv('movies.csv')

# Preprocessing: Convert categorical variables to numerical
label_encoder = LabelEncoder()
for column in data.columns:
    if data[column].dtype == 'object':
        data[column] = label_encoder.fit_transform(data[column])

# Impute missing values
imputer = SimpleImputer(strategy='mean')
data_imputed = imputer.fit_transform(data)

# Convert back to DataFrame
data_imputed = pd.DataFrame(data_imputed, columns=data.columns)

# Split the data into features (X) and target (y)
X = data_imputed.drop('rating', axis=1)  # Drop the target column
y = data_imputed['rating']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_

# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Create and train the SVM classifier
svm_classifier = SVC()
svm_classifier.fit(X_train_scaled, y_train)

# Evaluate the SVM classifier
y_pred_svm = svm_classifier.predict(X_test_scaled)
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print("Accuracy of SVM Classifier:", accuracy_svm)
print("Classification Report of SVM Classifier:")
print(classification_report(y_test, y_pred_svm))
```

```
Accuracy of SVM Classifier: 0.5384615384615384
Classification Report of SVM Classifier:
              precision    recall  f1-score   support

         1.0       0.00      0.00      0.00        35
         2.0       0.00      0.00      0.00         6
         3.0       0.88      0.12      0.21        59
         4.0       0.52      0.15      0.23       271
         5.0       0.60      0.29      0.39       422
         6.0       0.53      0.93      0.67       710
         8.0       0.00      0.00      0.00         3
        10.0       0.00      0.00      0.00        12
        11.0       0.00      0.00      0.00         1
        12.0       0.00      0.00      0.00        15

    accuracy                           0.54      1534
   macro avg       0.25      0.15      0.15      1534
weighted avg       0.53      0.54      0.47      1534
```

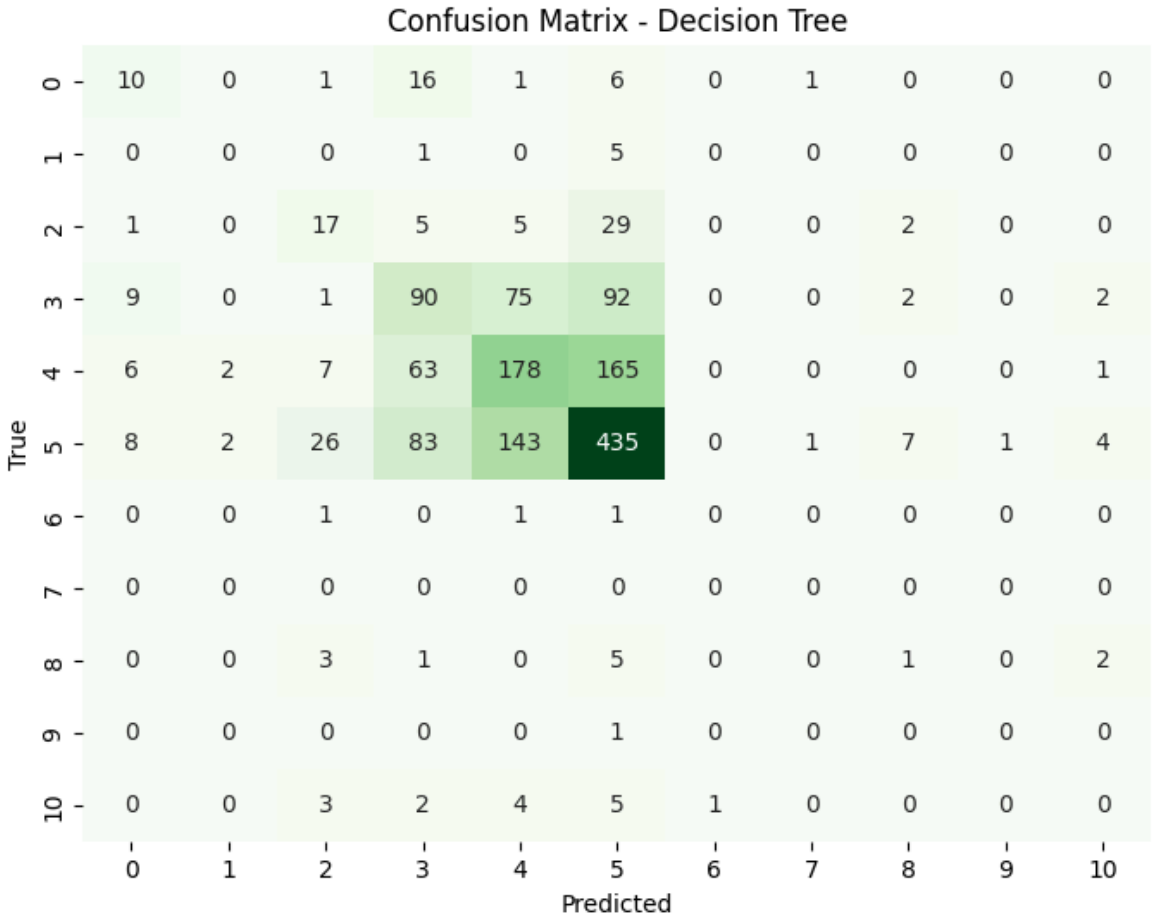
```
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
C:\Users\raksh\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2
kfra8p0\LocalCache\local-packages\Python312\site-packages\sklearn\metrics\_classi
fication.py:1509: UndefinedMetricWarning: Precision is ill-defined and being set
to 0.0 in labels with no predicted samples. Use `zero_division` parameter to cont
rol this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

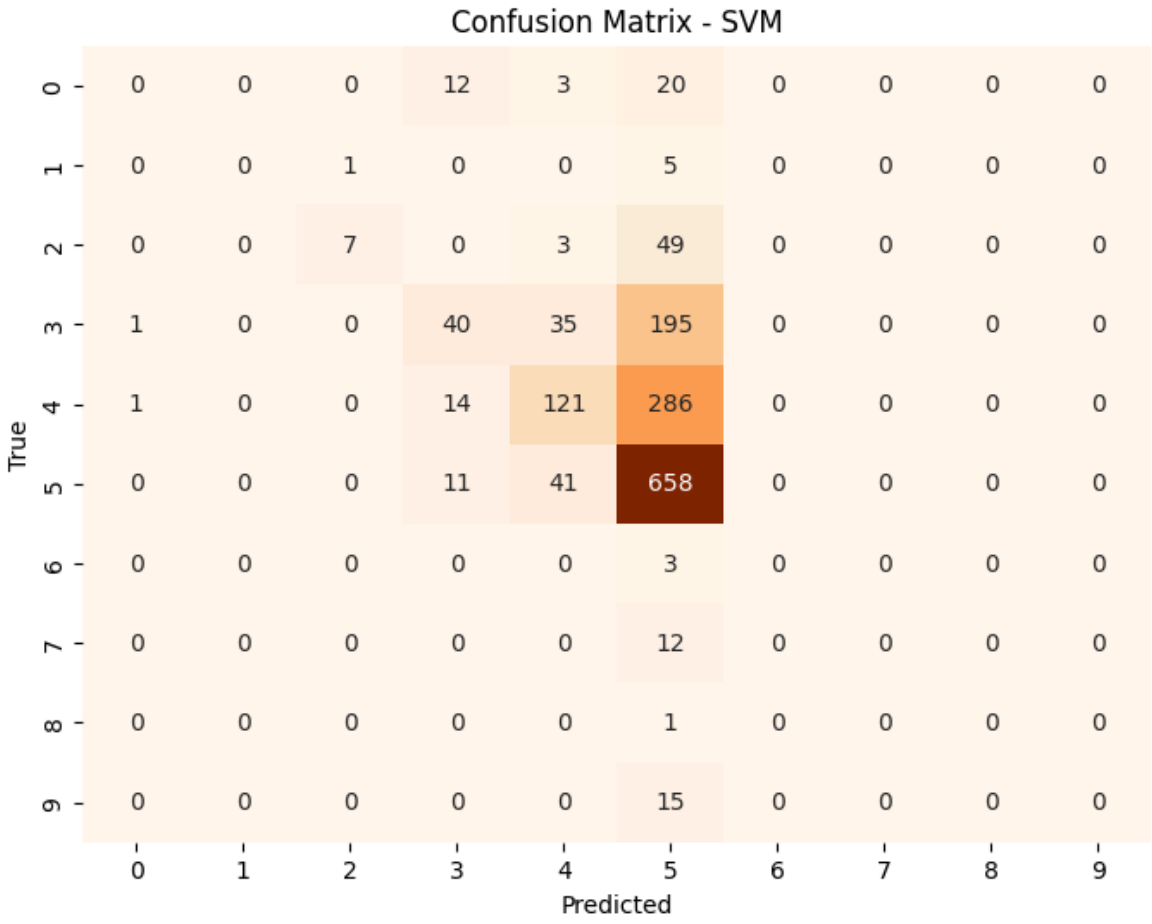
Confusion Matrix

In [30]:

```python
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt


# Compute confusion matrix for Decision Tree
cm_dt = confusion_matrix(y_test, y_pred_dt)

# Plot confusion matrix for Decision Tree
plt.figure(figsize=(8, 6))
sns.heatmap(cm_dt, annot=True, fmt='d', cmap='Greens', cbar=False)
plt.title('Confusion Matrix - Decision Tree')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()

# Compute confusion matrix for SVM
cm_svm = confusion_matrix(y_test, y_pred_svm)
```

```
# Plot confusion matrix for SVM
plt.figure(figsize=(8, 6))
sns.heatmap(cm_svm, annot=True, fmt='d', cmap='Oranges', cbar=False)
plt.title('Confusion Matrix - SVM')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```





Conclusion

Project Summary

In this project, we explored a dataset containing detailed information about movies released in 1980. Our goal was to analyze the data and build predictive models to estimate IMDb scores based on various features of the movies.

Key Steps Undertaken

1. **Data Preprocessing**: Handled missing values and converted categorical features to numerical values using Label Encoding.
2. **Exploratory Data Analysis (EDA)**: Conducted comprehensive EDA including:
   - Correlation heatmap to identify relationships between features.
   - Distribution plots to visualize the spread of numerical data.
   - Count plots to understand the frequency distribution of categorical variables.
3. **Machine Learning Models**: Implemented and evaluated:

Insights and Findings

- **Correlation Analysis**: Revealed significant relationships between features like budget, gross revenue, and IMDb scores.
- **Distribution Analysis**: Highlighted the skewness in budget and gross revenue, suggesting a few high-budget, high-grossing movies.
- **Genre and Rating**: Showed distinct patterns in how different genres and ratings influence movie scores.

Conclusion

Through this project, we have explored the dataset of movies released in 1980, performed EDA to uncover patterns and relationships, and built machine learning models to predict the IMDb scores of movies. The models provided insights into the factors influencing movie ratings and demonstrated the application of regression techniques in a real-world dataset.