IMDB SCORE PREDICTION

PROBLEM STATEMENT

- Introduction: IMDb (Internet Movie Database) is a popular platform for users to discover, rate, and review movies and television shows. IMDb provides a comprehensive database of films and TV series, along with user-generated ratings and reviews. Predicting IMDb scores is a valuable task in the entertainment industry, as it can help studios, filmmakers, and content creators gauge the potential success of their productions and make informed decisions.
- Problem Statement: The objective of this project is to develop a machine learning model that can accurately predict IMDb scores for movies or TV shows based on various features and attributes associated with the content. The problem can be framed as a regression task where the model will output a numeric score, representing the predicted IMDb rating for a given movie or show.
- Model Selection: The appropriate regression algorithm and optimizing its hyperparameters to achieve the best prediction performance.
- Evaluation metrics: Determining the most suitable evaluation metrics for assessing the model's accuracy, such as mean squared error (MSE), root mean squared error (RMSE), or R-squared (R2).ow.

- Methodology:
- Data Preprocessing: This involves cleaning the dataset, handling missing values, and encoding categorical features.
- Feature Selection and Engineering: Selecting relevant features and creating new ones if necessary to improve prediction performance.
- Model Selection: Choosing and implementing regression algorithms (e.g., linear regression, decision tree regression, random forest regression) for training and testing.
- Model Evaluation: Using appropriate evaluation metrics to assess the model's accuracy and generalization ability.
- Success Criteria: The success of this project will be determined by the model's ability to accurately predict IMDb scores, with a focus on minimizing prediction errors and achieving high correlation with the actual IMDb ratings.
- Stakeholders: Stakeholders for this project may include filmmakers, production studios, streaming platforms, and content creators interested in understanding the potential reception of their content.

DESIGN THINKING PROCESS

- Design Thinking is a problem-solving approach that emphasizes understanding user needs and developing innovative solutions. In the context of predicting IMDb scores, you can apply the Design Thinking process as follows:
- Empathize (Understand the Users and the Problem):Begin by understanding the stakeholders involved, such as filmmakers, studios, and IMDb users.
- Define (Clearly Define the Problem):Synthesize the information collected and define the problem more precisely. For example, define the specific goals and constraints of predicting IMDb scores.
- Ideate (Generate Solutions):Brainstorm potential solutions and approaches for predicting IMDb scores.Encourage creative thinking, and consider various data sources, features, and machine learning algorithms.
- Prototype (Create a Predictive Model):Develop a prototype predictive model based on the ideas generated in the previous step. Implement machine learning algorithms and regression models to build the predictive model.

- Test (Evaluate the Predictive Model):Evaluate the predictive model's performance using appropriate metrics (e.g., mean squared error, root mean squared error, R-squared).
- Iterate (Refine and Improve):Continuously refine the predictive model based on feedback and evaluation results.Explore different data sources and features to improve prediction accuracy.
- Implement (Deploy the Model):Provide access to the model for stakeholders who can benefit from IMDb score predictions, such as filmmakers and studios.
- Gather Feedback and Monitor:Continue to gather feedback and monitor the model's performance in a production environment.Collect user feedback to make ongoing improvements to the model, such as updating it with new data or features.
- Scale and Optimize:Consider scalability and optimization of the predictive model to handle a large volume of movie and TV show data.
- Celebrate Success and Share Learnings:Acknowledge and celebrate the successful implementation of the IMDb score prediction model. Regularly gather and incorporate feedback to ensure that the predictive model remains relevant and valuable to its stakeholders.

PHASE OF DEVELOPMENT

- The development of a system for predicting IMDb scores can be divided into several phases. These phases help organize the work, ensure a systematic approach, and facilitate project management. Here are the key phases for developing a predictive IMDb score system:
- Problem Definition: Clearly define the problem, objectives, and goals of predicting IMDb scores.
- Scope and Constraints: Define the scope of the project and any constraints, including data availability and timeline.
- Data Collection and Preparation:
- Data Sourcing: Gather relevant data sources, including movie attributes, IMDb ratings, and user reviews.
- Data Cleaning: Clean the data by handling missing values, outliers, and inconsistencies.
- Feature Selection and Engineering: Choose relevant features and create new ones if necessary to improve prediction accuracy.

- Model Development:
- Algorithm Selection: Choose appropriate regression algorithms or machine learning models for IMDb score prediction (e.g., linear regression, decision tree regression, random forest regression).
- Model Training: Train the selected model using the prepared dataset.
- Hyperparameter Tuning: Optimize model hyperparameters to improve performance.
- Cross-Validation: Evaluate the model's performance using cross-validation techniques to ensure robustness.
- Evaluation and Validation:
- Metric Selection: Define appropriate evaluation metrics (e.g., mean squared error, root mean squared error, R-squared) to assess model performance.
- Testing: Use a held-out test dataset to evaluate the model's predictive accuracy.
- User Feedback: Collect feedback from stakeholders and end-users to validate the model's usefulness.

- Monitoring and Maintenance:
- Feedback Loop: Establish a feedback mechanism to continuously gather user feedback and monitor model performance.
- Model Updates: Periodically update the model with new data and retrain it to improve accuracy.
- Error Handling: Implement error-handling mechanisms to deal with unexpected issues.
- User Training and Support:Provide training and support to stakeholders and users on how to use the IMDb score prediction system effectively.
- Documentation and Reporting:Document the development process, including data sources, model details, and best practices.Generate reports on the model's performance and share insights with stakeholders.
- Project Closure and Review:Review the project's objectives and deliverables to ensure they have been met.Conduct a project closure meeting to acknowledge the completion of the IMDb score prediction system.

DATASET

- **Download the dataset for Predicting IMDb scores using the following link**
- **Dataset link:**
- •https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores
- **Title:** Netflix Original Films IMDb Scores Dataset
- **Description:** This dataset likely contains information about movies produced or distributed by Netflix, focusing on their IMDb scores, which represent user ratings and reviews. The dataset may include the following fields:

1. **Movie Title:** The title of the Netflix original film.
2. **Release Year:** The year in which the film was released.
3. **IMDb Rating:** The IMDb score or rating for the film, reflecting user ratings and reviews.
4. **Genre:** The genre or genres to which the film belongs (e.g., drama, comedy, action).
5. **Director:** The director(s) of the film.

6.Cast: The main actors and actresses in the film.
7.Duration: The duration or runtime of the film in minutes.
8.Description/Plot: A brief summary or plot description of the movie.
9.Country: The country or countries associated with the film's production.
10.Language: The primary language of the film.
11.Awards/Nominations: Information about awards or nominations the film has received.
12.Production Company: The production company or companies involved in making the movie.

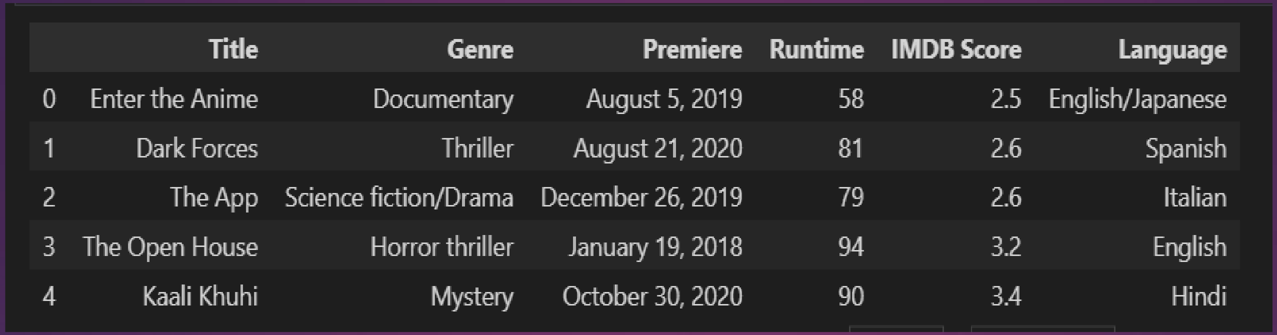
Potential Uses:

- Analyzing the distribution of IMDb scores for Netflix original films.
- Investigating the relationship between IMDb scores and other factors like genre, director, and cast.
- Understanding how IMDb scores may have changed over different release years.
- Identifying top-rated Netflix original films for recommendations.

DATASET INSERTION

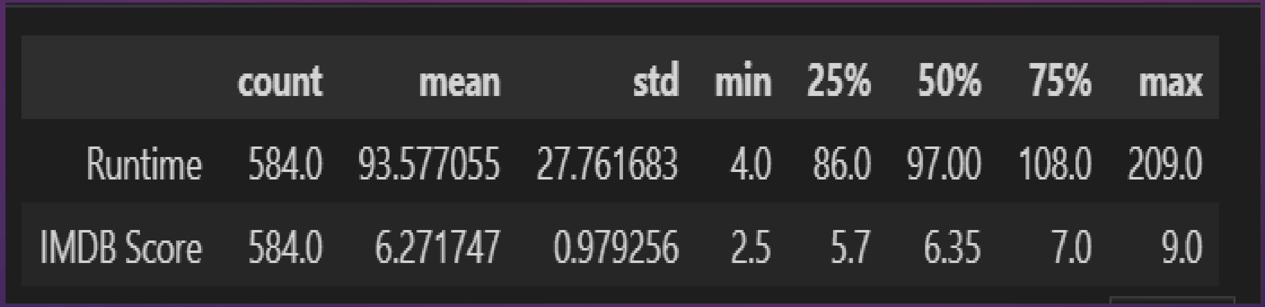
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from datetime import datetime,timedelta
ds = pd.read_csv(“imdb.csv")
ds_date = ds.copy()
ds.head(5)
```

OUTPUT



	Title	Genre	Premiere	Runtime	IMDB Score	Language
0	Enter the Anime	Documentary	August 5, 2019	58	2.5	English/Japanese
1	Dark Forces	Thriller	August 21, 2020	81	2.6	Spanish
2	The App	Science fiction/Drama	December 26, 2019	79	2.6	Italian
3	The Open House	Horror thriller	January 19, 2018	94	3.2	English
4	Kaali Khuhi	Mystery	October 30, 2020	90	3.4	Hindi

- ds.describe().T

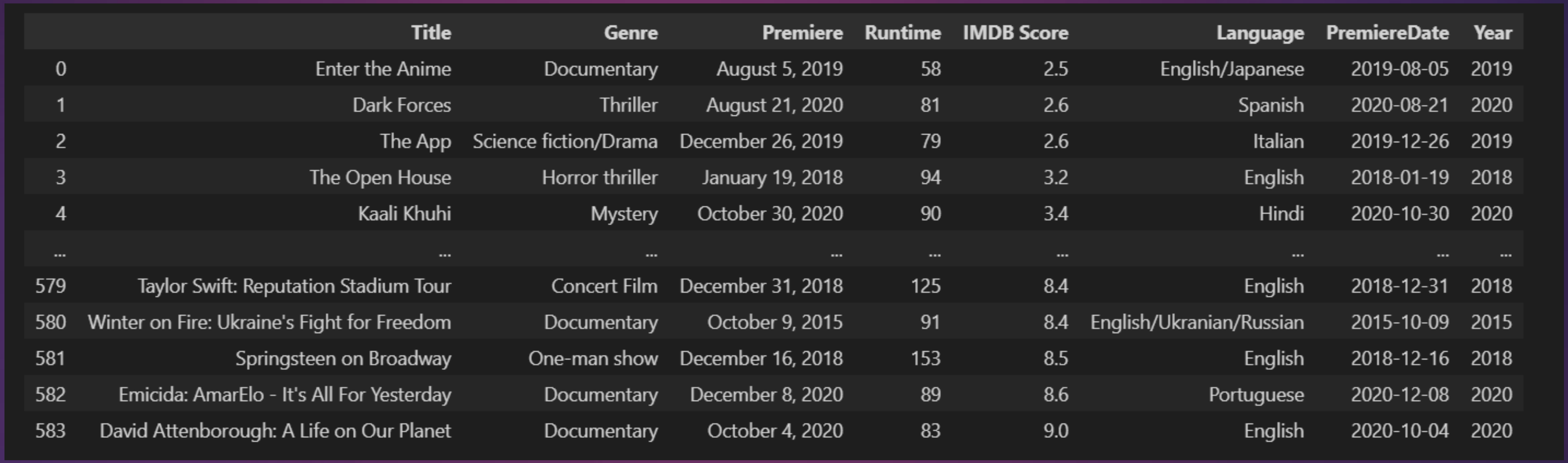


	count	mean	std	min	25%	50%	75%	max
Runtime	584.0	93.577055	27.761683	4.0	86.0	97.00	108.0	209.0
IMDB Score	584.0	6.271747	0.979256	2.5	5.7	6.35	7.0	9.0

DATASET PREMIERE

```
ds_date["Premiere"] = ds_date["Premiere"].apply(lambda x: "".join(x for x in x.replace(".",",")))
ds_date["PremiereDate"] = ds_date["Premiere"].apply(lambda x: datetime.strptime(x, "%B %d, %Y").date())
ds_date["Year"] = ds_date["Premiere"].apply(lambda x: "".join(x for x in x.replace(",","").split()[-1]))
ds_date["PremiereDate"] = pd.to_datetime(ds_date["PremiereDate"])
ds_date
```

OUTPUT

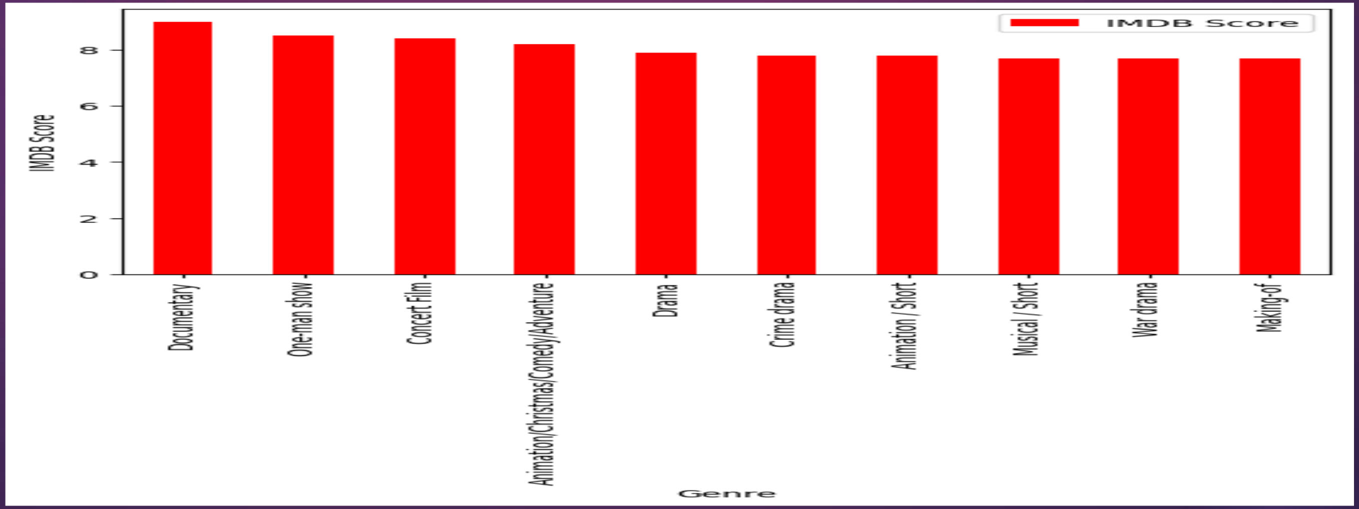


	Title	Genre	Premiere	Runtime	IMDB Score	Language	PremiereDate	Year
0	Enter the Anime	Documentary	August 5, 2019	58	2.5	English/Japanese	2019-08-05	2019
1	Dark Forces	Thriller	August 21, 2020	81	2.6	Spanish	2020-08-21	2020
2	The App	Science fiction/Drama	December 26, 2019	79	2.6	Italian	2019-12-26	2019
3	The Open House	Horror thriller	January 19, 2018	94	3.2	English	2018-01-19	2018
4	Kaali Khuhi	Mystery	October 30, 2020	90	3.4	Hindi	2020-10-30	2020
...	...	...	...	...	...	...	...	...
579	Taylor Swift: Reputation Stadium Tour	Concert Film	December 31, 2018	125	8.4	English	2018-12-31	2018
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary	October 9, 2015	91	8.4	English/Ukranian/Russian	2015-10-09	2015
581	Springsteen on Broadway	One-man show	December 16, 2018	153	8.5	English	2018-12-16	2018
582	Emicida: AmarElo - It's All For Yesterday	Documentary	December 8, 2020	89	8.6	Portuguese	2020-12-08	2020
583	David Attenborough: A Life on Our Planet	Documentary	October 4, 2020	83	9.0	English	2020-10-04	2020

IMDB SCORES

```
ds[['Genre', 'IMDB Score']].sort_values('IMDB Score',
ascending=False).drop_duplicates('Genre').head(10).plot(x='Genre',y='IMDB Score', kind='bar',
color='red')
•plt.xlabel('Genre')
•plt.ylabel('IMDB Score')
•plt.show(block=True)
```

OUTPUT



DATA PREPROCESSING

- Data preprocessing is a critical step when predicting IMDb scores or any other machine learning task. IMDb scores often refer to movie ratings on the Internet Movie Database, and predicting these scores could involve using various features related to movies.

1.Data Collection:

- Gather a dataset that includes information about movies and their IMDb scores. This data can be obtained from sources like IMDb's official website, Kaggle, or other movie-related datasets.

2.Data Cleaning:

- Remove duplicates: Check for and remove duplicate records, if any.
- Handle missing data: Address missing values in the dataset through methods like imputation, removal, or using domain-specific knowledge.
- Outlier detection: Identify and handle outliers in IMDb scores or other feature

3.Feature Selection:

- Choose relevant features: Select the features that are most likely to influence IMDb scores. These features may include genre, director, cast, runtime, budget, release year, etc.
- Remove irrelevant features: Eliminate features that are not likely to have a significant impact on IMDb scores.

4.Data Transformation:

- Encode categorical variables: Convert categorical features (e.g., genre, director) into numerical values using techniques like one-hot encoding or label encoding.
- Feature scaling: Normalize or standardize numerical features to have similar scales.
- Date/time feature extraction: Extract relevant information from date or time-related features, such as release date.

5.Feature Engineering:

- Create new features if needed: Generate additional features based on domain knowledge. For example, you could create a "sequel" feature to indicate if a movie is part of a series.
- Binning: Group continuous data into bins if it makes sense in the context of your analysis.

6.Data Splitting:

- Split the data into training and testing sets. This is essential for model evaluation.

7.Addressing Data Imbalance (if applicable):

- If the dataset is imbalanced (e.g., a disproportionate number of high or low IMDb scores), consider techniques like oversampling, undersampling, or using different evaluation metrics.

8.Scaling IMDb Scores (if needed):

- If IMDb scores are on a different scale (e.g., 1-10), you may need to rescale them to match the output of your predictive model.

9.Text Data Processing (if using movie descriptions or reviews):

- Tokenization: Break down textual data into words or tokens.

10.Model-Specific Preprocessing (if applicable):

Some machine learning models may require specific preprocessing steps. For example, sequence padding for recurrent neural networks (RNNs) when analyzing movie reviews.

11.Handling Data Leakage:

•Ensure that there is no data leakage, where information from the future (e.g., future IMDb scores) inadvertently influences predictions.

12.Standardization and Scaling:

•Standardize or scale the features to have a mean of 0 and a standard deviation of 1, which can be important for some machine learning algorithms.

13.Data Exploration and Visualization:

•Explore the data to gain insights and visualize relationships between features and IMDb scores. Visualization can help you understand the data better.

These data preprocessing steps are essential for building a predictive model to estimate IMDb scores accurately. Once the data is prepared, you can proceed to model selection, training, and evaluation to build a predictive IMDb score model.

MODEL TRAINING

1.Data Splitting:

- The data is first split into features (X) and the target variable (y).
  - X contains a single feature, 'Runtime,' which represents the runtime of movies.
  - y contains the target variable, 'IMDB Score,' which represents the IMDb scores of the movies.

2.Train-Test Split:

- The data is further split into training and testing sets using the train_test_split function from scikit-learn.
  - X_train and y_train represent the features and target variable for the training set.
  - X_test and y_test represent the features and target variable for the testing set.
  - The split is performed with a test size of 20% (test_size=0.2), and a random seed (random_state=42) is set to ensure reproducibility.

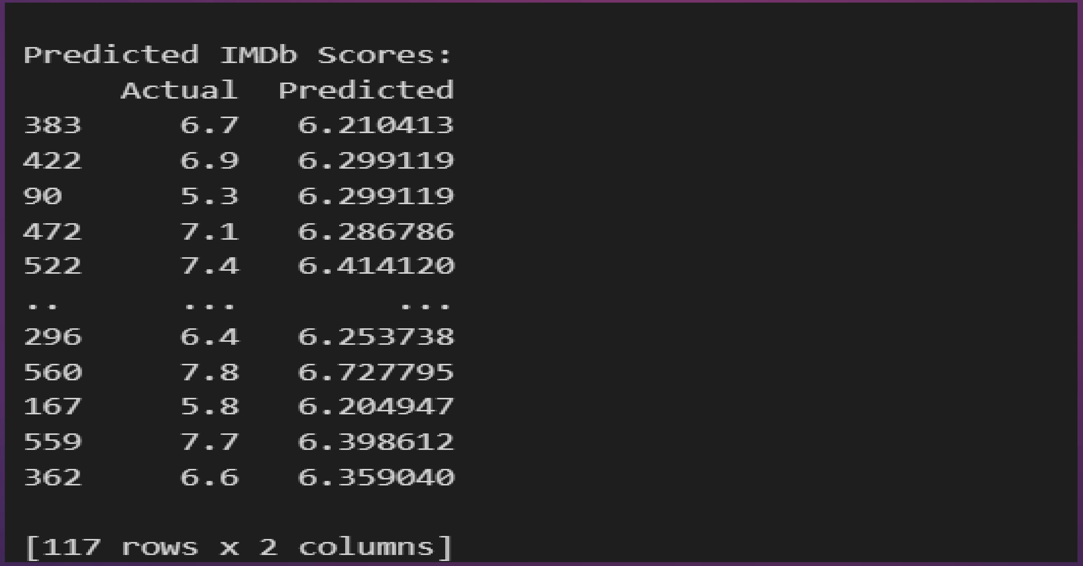
3.Scatter Plot Visualization:

- A scatter plot is created to visualize the relationship between 'Runtime' (X-axis) and 'IMDB Score' (Y-axis) for the training data.

PROGRAM

```
# Print the predicted IMDb scores
print("\nPredicted IMDb Scores:")
print(pd.DataFrame({'Actual': y_test, 'Predicted': y_pred}))
```

OUTPUT



```
Predicted IMDb Scores:
     Actual  Predicted
383     6.7   6.210413
422     6.9   6.299119
90      5.3   6.299119
472     7.1   6.286786
522     7.4   6.414120
..      ...        ...
296     6.4   6.253738
560     7.8   6.727795
167     5.8   6.204947
559     7.7   6.398612
362     6.6   6.359040

[117 rows x 2 columns]
```

Program for Mean Squared Error

```
# Print the Mean Squared Error
print(f"\nMeanSquared Error: {mse}")
```

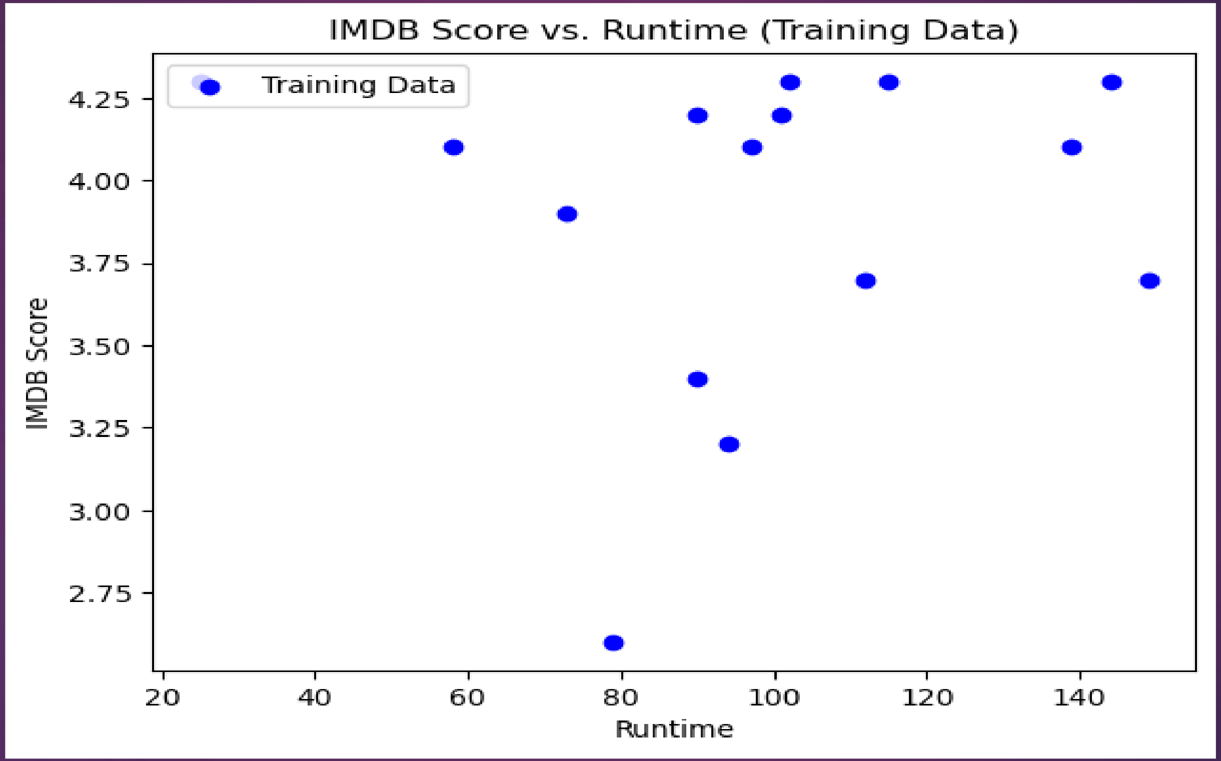
Output

```
Mean Squared Error: 0.9882182648225284
```

SCATTER PLOT FOR MODEL TRAINING

```
# Split the data into features (X) and target (y)
X = df[['Runtime']]
y = df['IMDB Score']
# 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_state=42)
# Create a scatter plot for model training
plt.scatter(X_train, y_train, color='blue', label='Training Data')
plt.xlabel("Runtime")
plt.ylabel("IMDB Score")
plt.legend(loc='upper left')
plt.title("IMDB Score vs. Runtime (Training Data)")
# Show the scatter plot
plt.show()
```

OUTPUT


IMDB Score vs. Runtime (Training Data)
Training Data
IMDB Score
4.25
4.00
3.75
3.50
3.25
3.00
2.75
Runtime
20
40
60
80
100
120
140

CHOICE OF REGRESSION ALGORITHM

The choice of a regression algorithm for predicting IMDb scores depends on various factors, including the characteristics of the dataset, the nature of the problem, and the specific goals of the prediction task.

1.**Data Characteristics:**

1. **Linear Relationships:** If the relationship between the input features (e.g., movie attributes) and IMDb scores is approximately linear, simple linear regression may be a suitable choice.
2. **Complex Relationships:** If the relationship is more complex, consider non-linear regression algorithms like decision tree regression, random forest regression, or support vector regression.

2.Feature Space:

1. **Feature Engineering:** The selection and engineering of features can significantly impact the choice of algorithm.
2. **Categorical Features:** If the dataset includes categorical features (e.g., movie genre, director, cast), you may need to encode them appropriately.

3.Model Complexity:

1. **Model Interpretability:** Consider the need for model interpretability. Linear regression provides a straightforward interpretation of how each feature impacts the IMDb score. If interpretability is a priority, linear regression may be preferable.
2. **Ensemble Models:** If you are open to using ensemble methods, random forest regression combines multiple decision trees and can provide both good predictive performance and feature importance analysis.

4.Performance Metrics:

1. Choose the algorithm based on the evaluation metrics that align with your project goals. Common metrics for regression tasks include mean squared error (MSE), root mean squared error (RMSE), and R-squared (R2). Different algorithms may perform better on different metrics.

5.Data Size and Computation:

1. Consider the size of the dataset. Linear regression can be more efficient and faster to train on large datasets, whereas complex models like deep learning may require significant computational resources.

6.Model Robustness and Generalization:

1. Assess how well the chosen algorithm generalizes to new, unseen data. Cross-validation and robustness tests can help determine which algorithm is more resilient to overfitting.

7.Handling Outliers and Noise:

1. Some regression algorithms are more robust to outliers and noisy data. Robust regression methods or ensemble models can be more resilient in such cases.

8.Scalability:

1. Consider the scalability of the algorithm. If the prediction system needs to handle a large volume of IMDb score predictions, ensure that the chosen algorithm is scalable for real-time or batch processing.

9.Experience and Domain Knowledge:

1. Personal or team experience with a specific algorithm can also influence the choice. Choose an algorithm that you or your team are proficient with and can effectively implement.

EVALUATION

1. **Splitting the Data:**
   1. The first step is to split the dataset into a training set and a testing set (or validation set). The training set is used to train the model, and the testing set is reserved for evaluation.
2. **Evaluation Metrics:**
   1. Choose appropriate evaluation metrics for regression tasks. Common metrics include:
      1. **Mean Squared Error (MSE):** It measures the average squared difference between predicted and actual IMDb scores. Lower values are better.
      2. **Root Mean Squared Error (RMSE):** It's the square root of the MSE and provides the error in the same units as the IMDb scores.
      3. **Mean Absolute Error (MAE):** It measures the average absolute difference between predicted and actual scores.
      4. **R-squared (R2):** It quantifies the proportion of the variance in IMDb scores that the model explains. A higher R2 indicates a better fit.
3. **Model Training and Testing:**
   1. Train the predictive model on the training set using the chosen algorithm and features.
   2. Test the model's performance on the testing set using the selected evaluation metrics.

4.Visualizations:

1. Create visualizations to help understand the model's performance. Scatter plots of predicted vs. actual IMDb scores can be useful for visual inspection. A scatter plot should show points closely following a diagonal line.

5.Cross-Validation:

1. Consider using k-fold cross-validation to assess the model's stability and generalization. Cross-validation involves splitting the dataset into k subsets, training the model on k-1 subsets, and testing it on the remaining subset. This process is repeated k times, and results are averaged.

6.Residual Analysis:

1. Analyze the residuals (the differences between predicted and actual IMDb scores). A well-fitted model should have residuals that are normally distributed and centered around zero.

7.Feature Importance:

1. If applicable, assess the importance of features in the model. Some algorithms provide feature importance scores, which can help identify which factors most strongly influence IMDb scores.

8.Comparative Analysis:

1. Compare the model's performance against baseline models or other regression algorithms to determine if your chosen model outperforms alternatives.

9.Bias and Fairness:

1. Investigate potential biases and fairness issues in the model's predictions. Ensure that the model is not disproportionately impacting certain groups or genres of movies.

10.User Feedback:

1. If possible, gather user feedback or feedback from stakeholders who use the IMDb score predictions. This feedback can provide insights into the practical utility and accuracy of the model.

11.Model Adjustments:

1. Based on the evaluation results, you may need to fine-tune hyperparameters, adjust the model's architecture, or make feature engineering changes to improve prediction performance.

12.Documentation and Reporting:

1. Document the evaluation results and the model's performance in a report. Share the findings and insights with relevant stakeholders.

PROGRAM

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
# Load the dataset (ensure data preprocessing as needed)file_path = "netflix.csv"
encoding = 'latin-1' # You can try 'cp1252' or other encodings if needed
# Create a DataFrame from the CSV file with the specified encoding
data = pd.read_csv(file_path, encoding=encoding)
# Feature Engineering: Use 'Genre' and 'Runtime' as features
X = data[['Genre', 'Runtime']]
y = data['IMDB Score']
# One-hot encode 'Genre'
X = pd.get_dummies(X, columns=['Genre'], drop_first=True)
```

```
Split the data into a training set and a testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Create and train a Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)
# Make predictions on the test set
y_pred = model.predict(X_test)
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error (MSE): {mse}")
print(f"R-squared
```

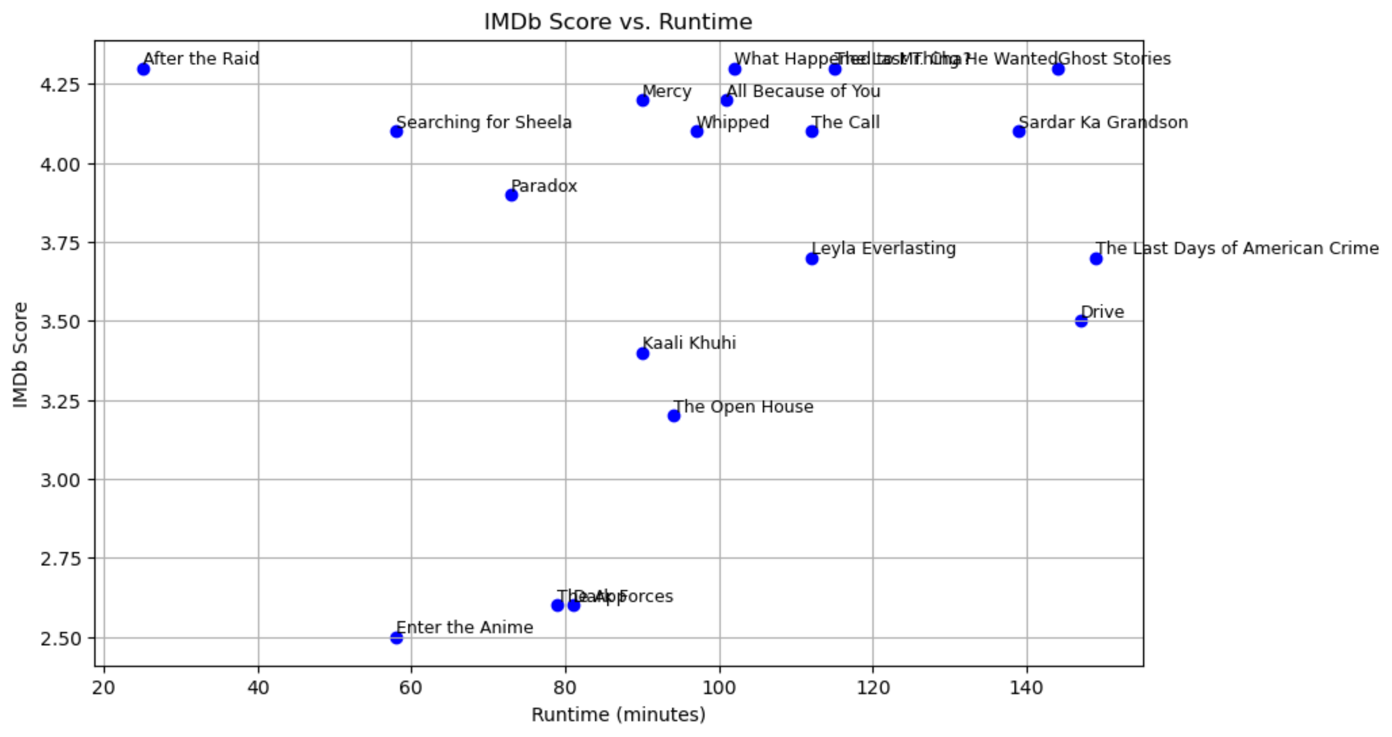
OUTPUT

```
Mean Squared Error (MSE): 0.9123928299593 R-squared
(R2) score: 0.12095921918523744
```

SCATTER PLOT FOR EVALUATION

```
# Create the scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(runtimes, imdb_scores, color='blue', marker='o')
plt.title('IMDb Score vs. Runtime')
plt.xlabel('Runtime (minutes)')
plt.ylabel('IMDb Score')
plt.grid(True)
# Annotate the points with movie titles
for i, title in enumerate(titles):
plt.annotate(title, (runtimes[i], imdb_scores[i]), fontsize=9,
verticalalignment='bottom')
plt.show()
```

OUTPUT


IMDb Score vs. Runtime
IMDb Score
Runtime (minutes)
4.25
4.00
3.75
3.50
3.25
3.00
2.75
2.50
20
40
60
80
100
120
140
After the Raid
Searching for Sheela
Paradox
Mercy
All Because of You
Whipped
What Happened to Mr. Cha?
The Last Thing He Wanted
Ghost Stories
The Call
Sardar Ka Grandson
Leyla Everlasting
The Last Days of American Crime
Drive
Kaali Khuhi
The Open House
The App
Dark Forces
Enter the Anime

CONCLUSION

- In conclusion, predicting IMDb scores is a valuable endeavor with various applications in the entertainment industry, including assisting filmmakers, studios, and movie enthusiasts in making informed decisions and recommendations. To successfully predict IMDb scores, a systematic approach is required, including data collection, model development, and rigorous evaluation.

- predicting IMDb scores is a complex task that requires a combination of data preparation, machine learning techniques, and careful evaluation. When executed effectively, this predictive capability can provide valuable insights and aid decision-making in the movie industry, benefiting both professionals and movie enthusiasts. However, it's essential to remember that IMDb scores are influenced by various factors, including user preferences, trends, and subjectivity, so predictions may not always perfectly reflect audience reception.