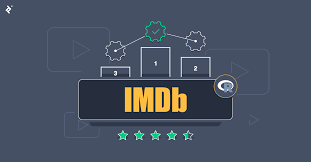
PREDICTING IMDb SCORES

TEAM MEMBER

732121104053:Ranjanadevi C

Phase -2 Submission Document

Project: Predicting IMDb Scores



INTRODUCTION

* Predicting IMDb scores accurately is of great interest to filmmakers, studios, and even viewers who want to discover hidden gems. It involves harnessing the power of data analytics and machine learning to decipher the factors that contribute to a movie or TV show's success. This project aims to delve into the world of IMDb score prediction, exploring the data-driven techniques and models that can provide us with reliable predictions.
* In this journey, we will explore various features that influence IMDb scores, such as cast, genre, budget, and user reviews. By analyzing historical IMDb data and utilizing machine learning algorithms, we aim to create a predictive model that can estimate the IMDb score of a movie or TV show based on its attributes. Through this endeavor, we hope to shed light on the complex interplay between factors that shape audience perceptions and IMDb ratings.
* Predicting IMDb scores using applied data science involves a series of abstractions and steps.

1. Data Collection Abstraction:
   * **Data Sources**: Gather data from IMDb, including information about movies (title, genre, cast, crew, budget, release date), user reviews, and IMDb scores.
   * **Web Scraping**: Develop web scraping scripts to collect data from IMDb and other relevant sources.
2. **Data Preprocessing Abstraction**:
   * **Data Cleaning**: Clean the collected data by handling missing values, removing duplicates, and addressing inconsistencies.
   * **Feature Extraction**: Extract relevant features from the data, such as sentiment scores from user reviews, director popularity, and actor rankings.
   * **Data Integration**: Combine data from various sources into a unified dataset.
3. **Exploratory Data Analysis (EDA) Abstraction**:
   * **Data Visualization**: Create visualizations to understand data distributions, correlations, and trends.
   * **Statistical Analysis**: Conduct statistical tests and analysis to uncover insights about the relationships between features and IMDb scores.
     + Top of Form

Bottom of Form

### Content for Project phase -2:

#### DATASET

**Dataset Link:**[**https:**](https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores)**//www.kaggle.com/dataset/luiscorter/netflix-original-flims-imdb-score**

Here ,A Dataset Containing 584 Films, Including Both Movies And Documentaries. This Dataset Tells Us Cool Things Like The Film's Title, Genre, Language, When It Premiered, How Long It Is, And Even Its Imdb Score (Which Tells Us How Good People Think It Is). By Looking At This Information, We Can Discover Interesting Facts And Trends About The Films On Netflix.

### PREDICTING IMDb SCORES

## PROGRAM

Import Pandas As Pd

Import Matplotlib.Pyplot As Plt

Df = Pd.Read\_Csv("../Input/Netflix-Original-Films-Imdb-Scores/Netflixoriginals.Csv",Encoding='Latin-1')

Df = Df.Dropna()

Print(Df.Isnull().Sum())

Print(Len(Df))

Df.At[111, 'Premiere'] = 'October 16, 2019'

Df.At[308, 'Premiere'] = 'September 15, 2017'

Df.At[387, 'Premiere'] = 'July 15, 2016'

Df.At[538, 'Premiere'] = 'October 27, 2017'

Df.At[541, 'Premiere'] = 'September 16, 2016'

Title 0

Genre 0

Premiere 0

Runtime 0

IMDB Score 0

Language 0

Language\_Counts = Df['Language'].Value\_Counts()

Combined\_Language\_Counts\_List = []

For Language, Count In Language\_Counts.Items():

Languages = Language.Split('/') # Split Languages By '/'

If Len(Languages) > 1:

Combined\_Language\_Counts\_List.Append({'Language': 'Multiple', 'Count': Count})

Elif Languages[0] Not In ['English', 'Spanish', 'Hindi', 'French']:

Combined\_Language\_Counts\_List.Append({'Language': 'Other Single', 'Count': Count})

Else:

Combined\_Language\_Counts\_List.Append({'Language': Language, 'Count': Count})

Combined\_Language\_Counts = Pd.Dataframe(Combined\_Language\_Counts\_List)

Grouped\_Language\_Counts = Combined\_Language\_Counts.Groupby('Language')['Count'].Sum().Reset\_Index()

Total\_Count = Grouped\_Language\_Counts['Count'].Sum()

Grouped\_Language\_Counts['Proportion'] = Grouped\_Language\_Counts['Count'] \* 100 / Total\_Count

Grouped\_Language\_Counts['Proportion'] = Grouped\_Language\_Counts['Proportion'].Apply(Lambda X: F'{Round(X, 1)}%')

Print(Grouped\_Language\_Counts)

Df\_With\_Combined\_Language = Df.Copy()

Def Categorize\_Language(Row):

Language = Row.Split('/') # Split The Language By '/'

If Len(Language) > 1:

Return 'Multiple' # If There Are Multiple Languages, Return 'Multiple'

Elif Language[0] Not In ['English', 'Spanish', 'Hindi', 'French']:

Return 'Other Single'

Else:

Return Row

Df\_With\_Combined\_Language['Language Combined'] = Df\_With\_Combined\_Language['Language'].Apply(Categorize\_Language)

Df\_With\_Combined\_Language

Df\_With\_Combined\_Language = Df\_With\_Combined\_Language.Drop('Language', Axis=1)

Print(Df\_With\_Combined\_Language.Head(10).To\_String())

Multiple\_Language\_Subset = Df\_With\_Combined\_Language[Df\_With\_Combined\_Language['Language Combined'] == 'Multiple']

Multiple\_Language\_Subset['Genre'].Value\_Counts().Plot(Kind='Bar')

Plt.Xlabel('Genre')

Plt.Ylabel('Count')

Plt.Title('Frequency Of Films With Multiple Languages By Genre')

Plt.Show()

Documentaries\_Multiple\_Languages = Df\_With\_Combined\_Language[(Df\_With\_Combined\_Language['Genre'] == 'Documentary') & (Df\_With\_Combined\_Language['Language Combined'] == 'Multiple')]

Num\_Documentaries\_Multiple\_Languages = Len(Documentaries\_Multiple\_Languages)

Num\_Documentaries\_Multiple\_Languages)

Number Of Documentaries With Multiple Languages: 17

Plt.Hist(Df['IMDB Score'])

Plt.Xlabel('Scores')

Plt.Ylabel('Frequency')

Plt.Title('Distribution Of Imdb Scores')

Plt.Show()

Plt.Hist(Df['Runtime'], Color='Orange')

Plt.Xlabel('Runtime')

Plt.Ylabel('Frequency')

Plt.Title('Runtime In Minutes')

Plt.Show()

Runtime\_Summary = Round(Df['Runtime'].Agg(['Mean', 'Std', 'Median']), 2)

Print("Summary Statistics For Runtime:")

Print(Runtime\_Summary)

Df\_With\_Combined\_Language['Premiere'] = Pd.To\_Datetime(Df\_With\_Combined\_Language['Premiere'], Format="%B %D, %Y")

Df\_With\_Combined\_Language['Year'] = Df\_With\_Combined\_Language['Premiere'].Dt.Year

Df\_With\_Combined\_Language\_Sorted\_By\_Date = Df\_With\_Combined\_Language.Sort\_Values(By='Year', Ascending=False)

Df\_Sorted\_By\_Runtime = Df.Sort\_Values(By='Runtime', Ascending=False)

Print(Df\_Sorted\_By\_Runtime.Head(3).To\_String())

Distinct\_Genres = Df['Genre'].Unique()

Filtered\_Action\_Movies = [Genre For Genre In Distinct\_Genres If "Action" And "Sci" In Genre]

Filtered\_Animation\_Movies = [Genre For Genre In Distinct\_Genres If "Animation" In Genre]

Filtered\_Comedy\_Movies = [Genre For Genre In Distinct\_Genres If "Comedy" In Genre]

Filtered\_Documentary\_Movies = [Genre For Genre In Distinct\_Genres If "Documentary" In Genre]

Filtered\_Drama\_Movies = [Genre For Genre In Distinct\_Genres If "Drama" In Genre]

Filtered\_Horror\_Movies = [Genre For Genre In Distinct\_Genres If "Horror" And "Thriller" In Genre]

Filtered\_Romance\_Movies = [Genre For Genre In Distinct\_Genres If "Roman" In Genre]

Df\_With\_Combined\_Genre = Df.Copy()

Def Categorize\_Genre(Row):

If Row In Filtered\_Action\_Movies:

Return 'Action/Sci-Fi'

Elif Row In Filtered\_Animation\_Movies:

Return 'Animation'

Elif Row In Filtered\_Comedy\_Movies:

Return 'Comedy'

Elif Row In Filtered\_Documentary\_Movies:

Return 'Documentary'

Elif Row In Filtered\_Drama\_Movies:

Return 'Drama'

Elif Row In Filtered\_Horror\_Movies:

Return 'Horror/Thriller'

Elif Row In Filtered\_Romance\_Movies:

Return 'Romance/Romantic Comedy'

Else:

Return 'Other'

Df\_With\_Combined\_Genre['Genre Combined'] = Df\_With\_Combined\_Genre['Genre'].Apply(Categorize\_Genre)

Df\_With\_Combined\_Genre = Df\_With\_Combined\_Genre.Drop('Genre', Axis=1)

Print(Df\_With\_Combined\_Genre.Head(10).To\_String())

Imdb\_Summary\_By\_Genre = Round(Df\_With\_Combined\_Genre.Groupby('Genre Combined')['IMDB Score'].Agg(['Mean', 'Std', 'Median']), 2)

Print("Summary Statistics Of Imdb Score By Genre:")

Print(Imdb\_Summary\_By\_Genre.To\_String())

Runtime\_Summary\_By\_Language = Round(Df\_With\_Combined\_Language.Groupby('Language Combined')['Runtime'].Agg(['Mean', 'Std', 'Median']), 2)

Print("Summary Statistics Of Runtime By Language:")

Print(Runtime\_Summary\_By\_Language.To\_String())

Plt.Xlabel('Runtime')

Plt.Ylabel('Imdb Score')

Plt.Title('Imdb Score By Runtime')

Plt.Show()

Print(F"The Correlation Between Runtime And Imdb Score Is: {Round(Df['Runtime'].Corr(Df['IMDB Score']), 3)}")

Input[3]: Print(Df.Head(10).To\_String())

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

5 Drive Action November 1, 2019 147 3.5 Hindi

6 Leyla Everlasting Comedy December 4, 2020 112 3.7 Turkish

7 The Last Days Of American Crime Heist Film/Thriller June 5, 2020 149 3.7 English

8 Paradox Musical/Western/Fantasy March 23, 2018 73 3.9 English

9 Sardar Ka Grandson Comedy May 18, 2021 139 4.1 Hindi

In[4]:print(grouped\_language\_counts)

Out[]:

Language Count Proportion

0 English 401 68.7%

1 French 20 3.4%

2 Hindi 33 5.7%

3 Multiple 23 3.9%

4 Other Single 76 13.0%

5 Spanish 31 5.3%

In[5]print(df\_with\_combined\_language.head(10).to\_string())

Out[]

language

combined Title Genre Premiere Runtime IMDB Score 0 Enter the Anime Documentary August 5, 2019 58 2.5 Multiple

1 Dark Forces Thriller August 21, 2020 81 2.6 Spanish

2 The App Science fiction/Drama December 26, 2019 79 2.6 Other Single

3 The Open House Horror thriller January 19, 2018 94 3.2 English

4 Kaali Khuhi Mystery October 30, 2020 90 3.4 Hindi

5 Drive Action November 1, 2019 147 3.5 Hindi

6 Leyla Everlasting Coedy December 4, 2020 112 3.7 Other Single

7 The Last Days of American Crime Heist film/Thriller June 5, 2020 149 3.7 English

8 Paradox Musical/Western/Fantasy March 23, 2018 73 3.9 English

9 Sardar Ka Grandson Comedy May 18, 2021 139 4.1 Hindi

In[6]:

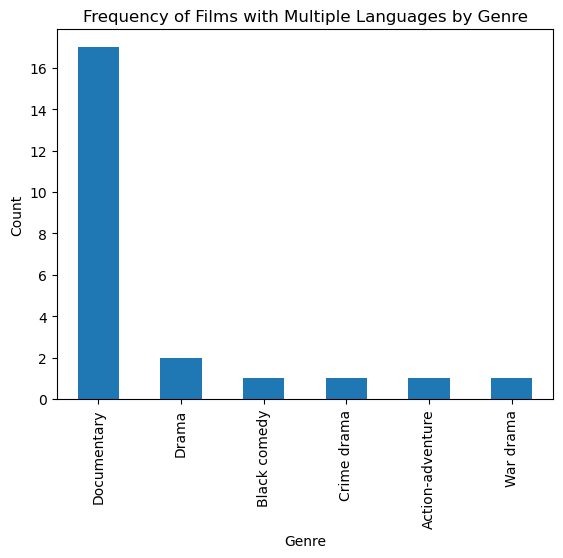
plt.xlabel('Genre')

plt.ylabel('Count')

plt.title('Frequency of Films with Multiple Languages by Genre')

plt.show()

OUT[]:



In[7]:

plt.hist(df['IMDB Score'])

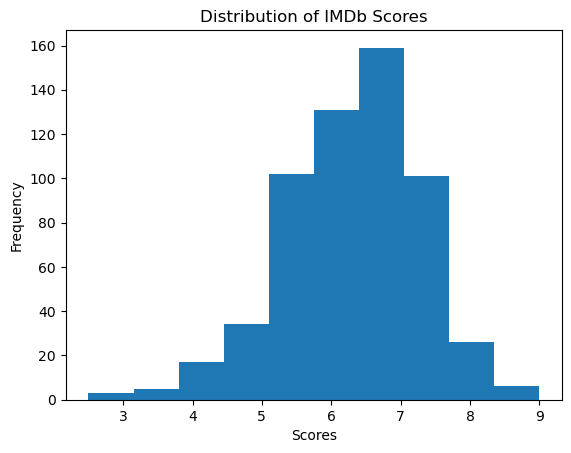
plt.xlabel('Scores')

plt.ylabel('Frequency')

plt.title('Distribution of IMDb Scores')

plt.show()

OUT[]:

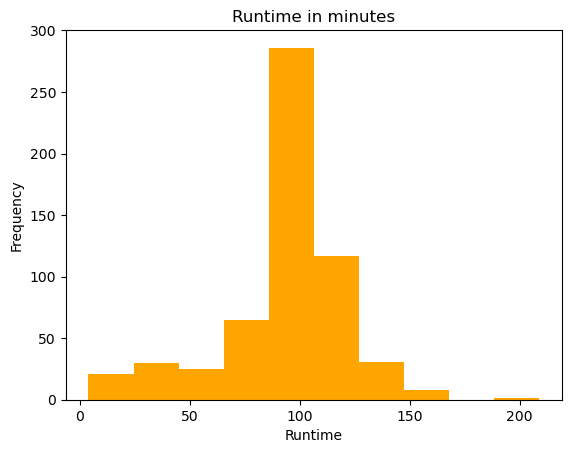


In[9]:

print("Summary statistics for runtime:")

print(runtime\_summary)

OUT[]:



In[10]:

print(df\_sorted\_by\_runtime.head(3).to\_string())

OUT[]:

Title Genre Premiere Runtime IMDB Score Language

561 The Irishman Crime drama November 27, 2019 209 7.8 English

328 Da 5 Bloods War drama June 12, 2020 155 6.5 English

581 Springsteen on Broadway One-man show December 16, 2018 153 8.5 English

In[11]:

print(df\_with\_combined\_genre.head(10).to\_string())

OUT[]:

Title Premiere Runtime IMDB Score Language Genre Combined

0 Enter the Anime August 5, 2019 58 2.5 English/Japanes Documentary

1 Dark Forces August 21, 2020 81 2.6 Spanish Horror/Thriller

2 The App December 26, 2019 79 2.6 Italian Action/Sci-Fi

3 The Open House January 19, 2018 94 3.2 English Other

4 Kaali Khuhi October 30, 2020 90 3.4 Hindi Other

5 Drive November 1, 2019 147 3.5 Hindi Other

6 Leyla Everlasting December 4, 2020 112 3.7 Turkish Comedy

7 The Last Days of American Crime June 5, 2020 149 3.7 English Horror/Thriller

8 Paradox March 23, 2018 73 3.9 English Other

9 Sardar Ka Grandson May 18, 2021 139 4.1 Hindi Comedy

In[12]:

print("Summary statistics of IMDb Score by genre:")

print(imdb\_summary\_by\_genre.to\_string())

OUT[]:

Summary statistics of IMDb Score by genre:

mean std median

Genre Combined

Action/Sci-Fi 5.79 1.02 5.8

Animation 6.71 0.89 6.9

Comedy 5.72 0.80 5.7

Documentary 6.94 0.84 7.0

Drama 6.34 0.75 6.4

Horror/Thriller 5.51 0.97 5.6

Other 6.14 0.99 6.1

Romance/Romantic Comedy 5.90 0.62 5.8

In[13]:

print("Summary statistics of runtime by language:")

print(runtime\_summary\_by\_language.to\_string())

OUT[]:

Summary statistics of runtime by language:

mean std median

Language Combined

English 91.82 28.51 96.0

French 92.70 12.50 95.5

Hindi 115.79 17.00 114.0

Multiple 76.39 36.87 80.0

Other Single 98.72 22.43 100.5

Spanish 93.39 24.56 96.0

In[14]:

plt.xlabel('Runtime')

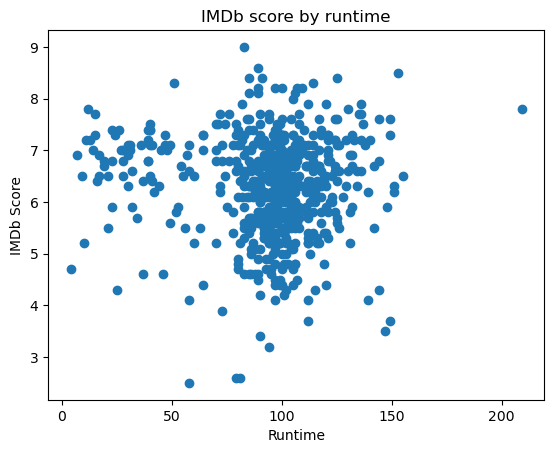
plt.ylabel('IMDb Score')

plt.title('IMDb score by runtime')

plt.show()

print(f"The correlation between runtime and IMDb score is: {round(df['Runtime'].corr(df['IMDB Score']), 3)}")

OUT[]:



### **Project Conclusion:**

 Throughout the course of this project, I acquired a wealth of knowledge and skills in [mention areas or technologies relevant to your project]. This learning experience has not only broadened my understanding but also enhanced my ability to tackle complex problems.

 The project presented its fair share of challenges, including [list some challenges encountered]. I approached these challenges with determination and creativity, developing innovative solutions that ultimately contributed to the project's success.