

IPREDICTING IMDB SCORES - ADS Phase5

2 TEAM NUMBER : 01

3.1 Final Submission

```
[6]: #importing necessary libraries import pandas as pd
from sklearn.preprocessing import StandardScaler,
LabelEncoder from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
#importing the netflix dataset
file_path =
r"C:\Users\Saranya\Desktop\IBM\NetflixOriginals.csv"
encoding = "ISO-8859-1" df = pd.read_csv(file_path,
encoding=encoding) df
```

```
[6]:
```

	Title	Genre \
0	Enter the Anime	Documentary
1	Dark Forces	Thriller
2	The App	Science fiction/Drama
3	The Open House	Horror thriller
4	Kaali Khuhi	Mystery
...
579	Taylor Swift: Reputation Stadium Tour	Concert Film
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary
581	Springsteen on Broadway	One-man show
582	Emicida: AmarElo - It's All For Yesterday	Documentary
583	David Attenborough: A Life on Our Planet	Documentary

	Premiere	Runtime	IMDB Score	Language
0	August 5, 2019	58	2.5	English/Japanese
1	August 21, 2020	81	2.6	Spanish
2	December 26, 2019	79	2.6	Italian
3	January 19, 2018	94	3.2	English

```

4    October 30, 2020    90    3.4    Hindi
..
579 December 31, 2018    125    8.4    English
580    October 9, 2015    91    8.4    English/Ukranian/Russian
581 December 16, 2018    153    8.5    English
582 December 8, 2020    89    8.6    Portuguese
583    October 4, 2020    83    9.0    English

```

[584 rows x 6 columns]

```
[7] d info()
```

```

<class
'pandas.core.frame.DataFrame'>
RangeIndex: 584 entries, 0 to
583 Data columns (total 6
columns):
#   Column      Non-Null Count  Dtype
---  -
0    Title 584 non-null    object
1    Genre 584 non-null    object
2    Premiere 584 non-null    object
3    Runtime 584 non-null    int64
4    IMDB Score 584 non-null    float64
5    Language 584 non-null    object dtypes: float64(1), int64(1),
object(4) memory usage: 27.5+ KB

```

```
[8] d head()
```

```

[8]:
      Title      Genre      Premiere Runtime \
0 Enter the Anime    Documentary  August 5, 2019    58
1   Dark Forces      Thriller  August 21, 2020    81
2   The App Science fiction/Drama  December 26, 2019    79
3 The Open House    Horror thriller  January 19, 2018    94
4   Kaali Khuhi      Mystery  October 30, 2020    90
   IMDB Score      Language
0    2.5 English/Japanese
1    2.6 Spanish 2  2.6 Italian 3  3.2
   English
4    3.4 Hindi

```

```
[9] #to display null values
d isnull()
```

```
[9]:      Title Genre Premiere Runtime IMDB Score Language
0    False False      False      False      False      False
1    False False      False      False      False      False
2    False False      False      False      False      False
3    False False      False      False      False      False
4    False False      False      False      False      False
..      ...      ...      ...      ...      ...
579  False False      False      False      False      False
580  False False      False      False      False      False
581  False False      False      False      False      False
582  False False      False      False      False      False
583  False False      False      False      False      False
[5  rows x 6 columns]
```

```
[10]: #handling null values

d fillna(df mean(), inplace Tr
d dropna(inplace Tr
```

```
[11]: #Display distinct languages
value_lang d Language value_counts()
prin \ Distinct languages:
prin value_lang
```

Distinct languages:

English	401
Hindi	33
Spanish	31
French	20
Italian	14
Portuguese	12
Indonesian	9
Japanese	6
Korean	6
German	5
Turkish	5
English/Spanish	5
Polish	3
Dutch	3
Marathi	3
English/Hindi	2
Thai	2
English/Mandarin	2
English/Japanese	2
Filipino	2

```

English/Russian      1
Bengali              1
English/Arabic       1
English/Korean       1
Spanish/English      1
Tamil                1
English/Akan         1
Khmer/English/French 1
Swedish              1
Georgian             1
Thia/English         1
English/Taiwanese/Mandarin 1
English/Swedish      1
Spanish/Catalan      1
Spanish/Basque       1
Norwegian            1
Malay                1
English/Ukranian/Russian 1
Name: Language, dtype:
int64

```

```

[12]: distinct_lang d Language unique()
      prin distinct lang

```

```

['English/Japanese' 'Spanish' 'Italian' 'English' 'Hindi' 'Turkish'
 'Korean' 'Indonesian' 'Malay' 'Dutch' 'French' 'English/Spanish'
 'Portuguese' 'Filipino' 'German' 'Polish' 'Norwegian' 'Marathi'
 'Thai'
 'Swedish' 'Japanese' 'Spanish/Basque' 'Spanish/Catalan'
 'English/Swedish'
 'English/Taiwanese/Mandarin' 'Thia/English' 'English/Mandarin'
 'Georgian'
 'Bengali' 'Khmer/English/French' 'English/Hindi' 'Tamil'
 'Spanish/English' 'English/Korean' 'English/Arabic'
 'English/Russian'
 'English/Akan' 'English/Ukranian/Russian']

```

```

[13]: #label encoder for language column

```

```

label_encoder  LabelEncoder()
d Language     label_encoder fit_transform(df[ Language ])
d

```

```

[13]:
           Titl           Genre \
0 Enter the Anime Documentary 1 Dark Forces Thriller
2                               The App Science fiction/Drama
3           The Open House     Horror
                                   thriller

```

4	Kaali Khuhi	Mystery
..
579	Taylor Swift: Reputation Stadium Tour	Concert Film
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary
581	Springsteen on Broadway	One-man show
582	Emicida: AmarElo - It's All For Yesterday	Documentary
583	David Attenborough: A Life on Our Planet	Documentary

Premiere Runtime IMDB Score Language

0	August 5, 2019	58	2.5	6
1	August 21, 2020	81	2.6	29
2	December 26, 2019	79	2.6	20
3	January 19, 2018	94	3.2	2
4	October 30, 2020	90	3.4	18
..

579	December 31, 2018	125	8.4	2
580	October 9, 2015	91	8.4	13
581	December 16, 2018	153	8.5	2
582	December 8, 2020	89	8.6	28
583	October 4, 2020	83	9.0	2

[5 rows x 6 columns]

[14]: *#scaling*

```

scale      StandardScaler()
df[Runtime] = scale.fit_transform(df[Runtime].values.reshape(-1, 1))
df

```

[14]:	Titl	Genre \
0	Enter the Anime Documentary	1 Dark Forces Thriller
2	The App Science fiction/Drama	
3	The Open House	Horror thriller
4	Kaali Khuhi	Mystery
..
579	Taylor Swift: Reputation Stadium Tour	Concert Film
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary

581	Springsteen on Broadway	One-man show			
582	Emicida: AmarElo - It's All For Yesterday	Documentary			
583	David Attenborough: A Life on Our Planet	Documentary			
Premiere Runtime IMDB Score Language					
0	August 5, 2019	-1.282615	2.5	6	
1	August 21, 2020	-0.453425	2.6	29	
2	December 26, 2019	-0.525528	2.6	20	
3	January 19, 2018	0.015248	3.2	2	
4	October 30, 2020	-0.128959	3.4	18	

579	December 31, 2018	1.132852	8.4	2	
580	October 9, 2015	-0.092907	8.4	13	
581	December 16, 2018	2.142301	8.5	2	
582	December 8, 2020	-0.165011	8.6	28	
583	October 4, 2020	-0.381321	9.0	2	

[5] rows x 6 columns

```
[15]: #train_test split

#X = df.drop('IMDB Score', axis=1)
#y = df['IMDB Score']

#X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)

[16]: #print("\n X_test info")
    #print(X_test.info())

[17]: # Drop non-numeric columns
    d = df.drop(['IMDB Score', 'Title', 'Genre', 'Premiere'], axis=1)
    d = d[['IMDB Score']]

# Import necessary libraries for model training and evaluation
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# 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_state=42)

# Initialize the Linear Regression model
model = LinearRegression()

# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model
m = mean_absolute_error(y_test, y_pred)
m = mean_squared_error(y_test, y_pred)
rm = mean_squared_error(y_test, y_pred, squared=False)
r = r2_score(y_test, y_pred)

print('Mean Absolute Error (MAE):', m)
print('Mean Squared Error (MSE):', m)
print('Root Mean Squared Error (RMSE):', rm)

print('R-squared (R2):', r)
```

Mean Absolute Error (MAE): 0.8066643972186746

Mean Squared Error (MSE): 0.9998118486476895
Root Mean Squared Error (RMSE): 0.999905919898312
R-squared (R2): 0.036735757620628084

```
[18]: pip install matplotlib
```

```
Requirement already satisfied: matplotlib in  
c:\users\saranya\anaconda3\lib\site-packages (3.5.1)  
Requirement already satisfied: fonttools>=4.22.0 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(4.25.0) Requirement already satisfied: python-dateutil>=2.7 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(2.8.2) Requirement already satisfied: cyclor>=0.10 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(0.11.0) Requirement already satisfied: pyparsing>=2.2.1 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(3.0.4) Requirement already satisfied: kiwisolver>=1.0.1 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(1.3.2) Requirement already satisfied: pillow>=6.2.0 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(9.0.1) Requirement already satisfied: packaging>=20.0 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib) (21.3)  
Requirement already satisfied: numpy>=1.17 in  
c:\users\saranya\anaconda3\lib\site-packages (from matplotlib)  
(1.23.5) Requirement already satisfied: six>=1.5 in  
c:\users\saranya\anaconda3\lib\site-  
packages (from python-dateutil>=2.7->matplotlib) (1.16.0)  
Note: you may need to restart the kernel to use updated packages.
```

```
[19]: # Import necessary libraries for model training and  
evaluation from sklearn.linear_model import
```

```
LinearRegression  
from sklearn.metrics import mean_absolute_error, mean_squared_error,  
r2_score
```

```
[20]: # 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_state=42)
```

```
[21]: # Initialize the Linear Regression  
model model = LinearRegression()  
  
# Train the model on the training  
data model.fit(X_train, y_train)  
  
# Make predictions on the test data  
y_pred = model.predict(X_test)
```

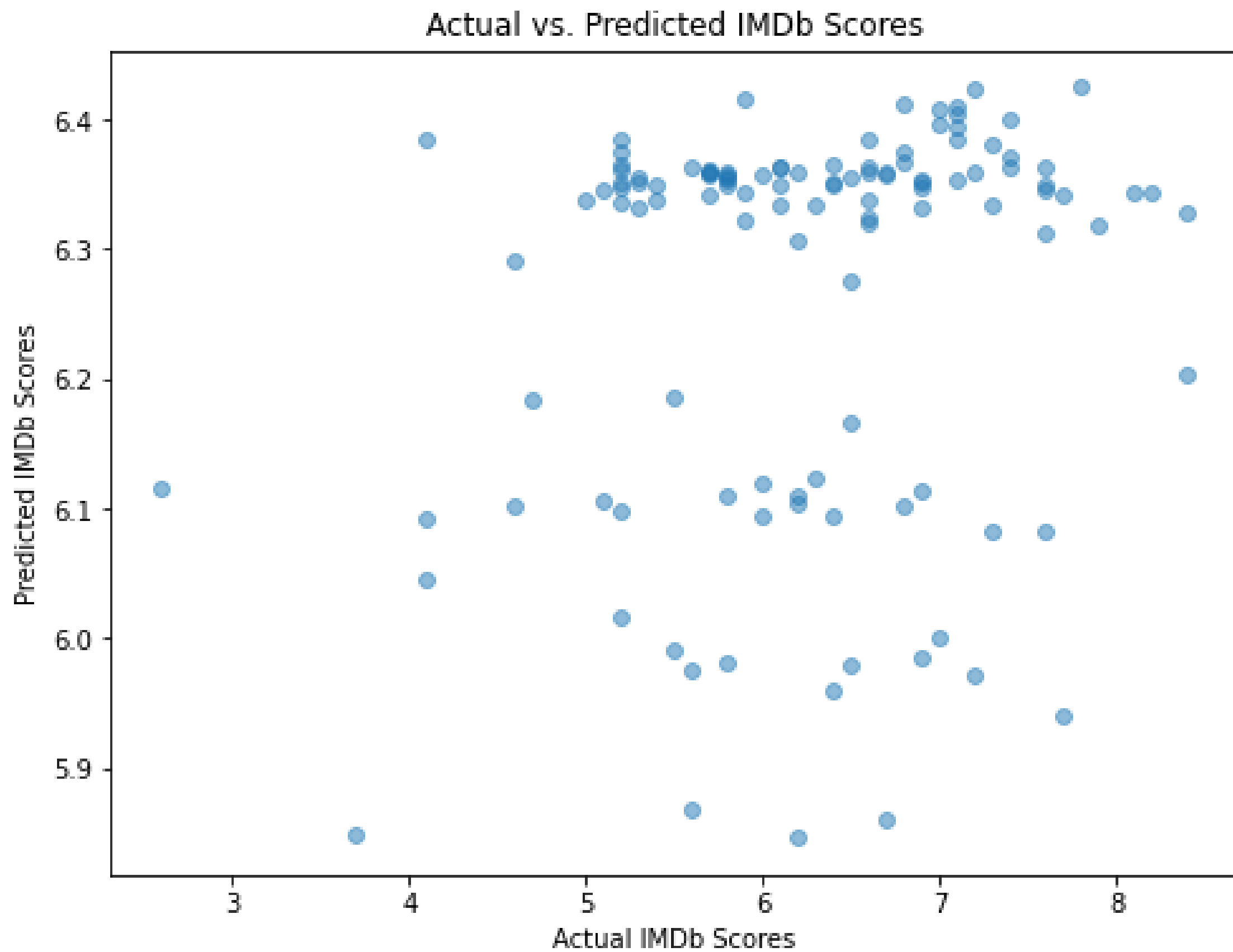


```
[22]: # Evaluate the model mae =  
mean_absolute_error(y_test, y_pred) mse =  
mean_squared_error(y_test, y_pred) rmse =  
mean_squared_error(y_test, y_pred,  
squared=False) r2 = r2_score(y_test, y_pred)  
  
print(f"Mean Absolute Error (MAE): {mae}")  
print(f"Mean Squared Error (MSE): {mse}")  
print(f"Root Mean Squared Error (RMSE):  
{rmse}") print(f"R-squared (R2): {r2}")
```

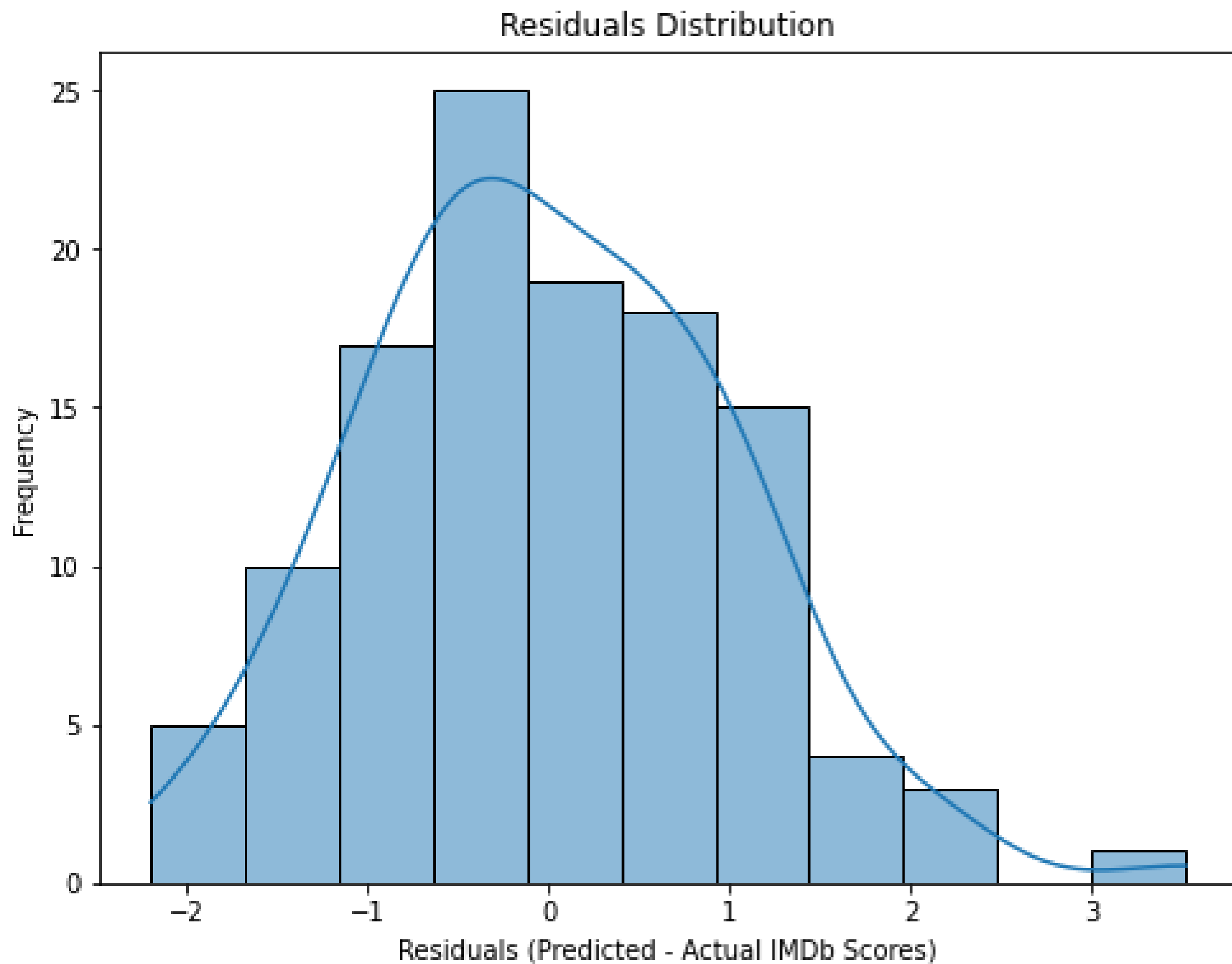
Mean Absolute Error (MAE): 0.8066643972186746
Mean Squared Error (MSE): 0.9998118486476895
Root Mean Squared Error (RMSE): 0.999905919898312
R-squared (R2): 0.036735757620628084

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

```
[24]: # Scatter plot of actual IMDb scores vs. predicted IMDb scores  
p = plt.figure(figsize=(10, 10))  
p = plt.scatter(y_test, y_pred, alpha=0.5)  
p = plt.xlabel('Actual IMDb Scores')  
p = plt.ylabel('Predicted IMDb Scores')  
p = plt.title('Actual vs. Predicted IMDb Scores')  
p = plt.show()
```



```
[25]: # Distribution plot of the residuals (predicted - actual
      # IMDb scores)
      residuals = y_pred - y_test
      plt.figure(figsize=(8, 6))
      sns.histplot(residuals,
                    kde=True)
      plt.xlabel("Residuals (Predicted - Actual IMDb Scores)")
      plt.ylabel("Frequency")
      plt.title("Residuals Distribution")
      plt.show()
```



```
[26]: from sklearn.ensemble import RandomForestRegressor
      from sklearn.model_selection import train_test_split
      from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
      import matplotlib.pyplot as plt
      import seaborn as sns
```

```
[27]: X = df.drop(["IMDB Score", "Title", "Genre", "Premiere"],
      axis=1)
      y = df["IMDB Score"]

      X_train, X_test, y_train, y_test = train_test_split(X, y,
      test_size=0.2, random_state=42)
```

```
[28]: model = RandomForestRegressor(random_state=42)
      model.fit(X_train, y_train)
```

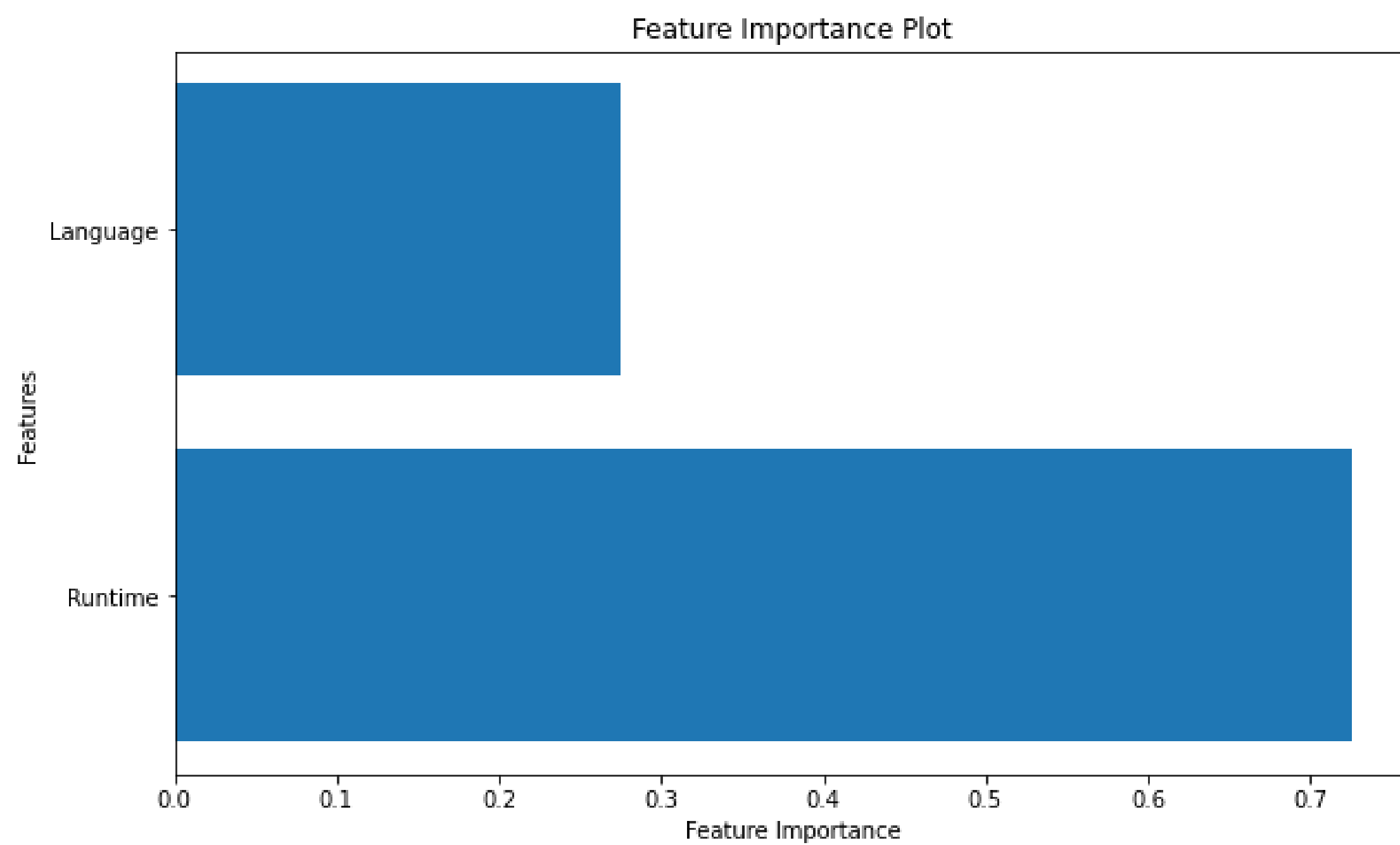
```
[28]: RandomForestRegressor(random_state=42)
```

```
[29]: if isinstance(model, RandomForestRegressor):
      feature_importance = model.feature_importances_
      feature_names = X_train.columns
```

```

p  figure(figsize = 10, 10)
p  barh(feature_names, feature_importance)
p  xlabel  Feature Importance
p  ylabel  Features
p  title  Feature Importance Plot
p  show()

```



```
[30]: d predict_imdb_score title, genre, premiere, runtime, language )
      # Create input data for prediction
      input_data = pd.DataFrame({ 'Runtime' : runtime, 'Language' : label_encoder
transform([language])})

      # Make a prediction
      predicted_imdb_score = model.predict(input_data)

      return predicted_imdb_score
```

```
[34]: # Example usage:
      titl      2023-01-01
      genr      Actio
      premiere  2023-01-01 # You can format the date accordingly
      runtim    4
      language  Englis

      predicted_score = predict_imdb_score(title, genre, premiere, runtime, language)
      print(f"Predicted IMDb Score: {predicted_score:.2f}")
```

Predicted IMDb Score: 7.31

```
[ ]: # User input title = input("Enter the movie title: ")
      genre = input("Enter the movie genre: ") premiere =
input("Enter the premiere date (YYYY-MM-DD): ") runtime
= float(input("Enter the movie runtime (in minutes):
")) language = input("Enter the movie language: ")

      predicted_score = predict_imdb_score(title, genre, premiere, runtime,
language) print(f"Predicted IMDb Score: {predicted_score:.2f}")
```