Predicting IMDb Scores

Phase – 4

Development part -2

ABINESH.M

(Team Member)

Explanation for Development part -I:

Step 1: Importing the required libraries and loading the dataset

In [57]:
```
#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
```

Out[57]:

	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
...	...	...	...	...	...	...
579	Taylor Swift: Reputation Stadium Tour	Concert Film	December 31, 2018	125	8.4	English
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary	October 9, 2015	91	8.4	English/Ukranian/Russian
581	Springsteen on Broadway	One-man show	December 16, 2018	153	8.5	English
582	Emicida: AmarElo - It's All For Yesterday	Documentary	December 8, 2020	89	8.6	Portuguese
583	David Attenborough: A Life on Our Planet	Documentary	October 4, 2020	83	9.0	English

584 rows × 6 columns

Step 2: Handling Missing Data

In [60]:
```
#to display null values
df.isnull()
```

Out[60]:

	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

584 rows × 6 columns

Handling the missing data

```
In [61]: #handling null values

         df.fillna(df.mean(), inplace=True)
         df.dropna(inplace=True)
```

Step 3: Identifying distinct languages

```
In [63]: distinct_lang = df['Language'].unique()
         print(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']
```

Step 4: Label encoder for language column

```
In [64]: #label encoder for language column

         label_encoder = LabelEncoder()
         df['Language'] = label_encoder.fit_transform(df['Language'])
         df
```

Out[64]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
0	Enter the Anime	Documentary	August 5, 2019	58	2.5	6
1	Dark Forces	Thriller	August 21, 2020	81	2.6	29
2	The App	Science fiction/Drama	December 26, 2019	79	2.6	20
3	The Open House	Horror thriller	January 19, 2018	94	3.2	2
4	Kaali Khuhi	Mystery	October 30, 2020	90	3.4	18
...	...	...	...	...	...	...
579	Taylor Swift: Reputation Stadium Tour	Concert Film	December 31, 2018	125	8.4	2
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary	October 9, 2015	91	8.4	13
581	Springsteen on Broadway	One-man show	December 16, 2018	153	8.5	2
582	Emicida: AmarElo - It's All For Yesterday	Documentary	December 8, 2020	89	8.6	28
583	David Attenborough: A Life on Our Planet	Documentary	October 4, 2020	83	9.0	2

584 rows × 6 columns

Step 5: Feature Scaling using StandardScaler

In [66]:
```
#scaling

scaler = StandardScaler()
df['Runtime'] = scaler.fit_transform(df['Runtime'].values.reshape(-1, 1))
df
```

Out[66]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
0	Enter the Anime	Documentary	August 5, 2019	-1.282615	2.5	6
1	Dark Forces	Thriller	August 21, 2020	-0.453425	2.6	29
2	The App	Science fiction/Drama	December 26, 2019	-0.525528	2.6	20
3	The Open House	Horror thriller	January 19, 2018	0.015248	3.2	2
4	Kaali Khuhi	Mystery	October 30, 2020	-0.128959	3.4	18
...	...	...	...	...	...	...
579	Taylor Swift: Reputation Stadium Tour	Concert Film	December 31, 2018	1.132852	8.4	2
580	Winter on Fire: Ukraine's Fight for Freedom	Documentary	October 9, 2015	-0.092907	8.4	13
581	Springsteen on Broadway	One-man show	December 16, 2018	2.142301	8.5	2
582	Emicida: AmarElo - It's All For Yesterday	Documentary	December 8, 2020	-0.165011	8.6	28
583	David Attenborough: A Life on Our Planet	Documentary	October 4, 2020	-0.381321	9.0	2

584 rows × 6 columns

Step 6: Splitting the data into a training set and a test

In [68]:
```
#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)
```

In [72]:
```
print("\n X_test info")
print(X_test.info())
```

```
 X_test info
<class 'pandas.core.frame.DataFrame'>
Int64Index: 117 entries, 383 to 362
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Title     117 non-null    object 
 1   Genre     117 non-null    object 
 2   Premiere  117 non-null    object 
 3   Runtime   117 non-null    float64
 4   Language  117 non-null    int32  
dtypes: float64(1), int32(1), object(3)
memory usage: 5.0+ KB
None
```

Feature Scaling, Model Training, and Evaluation Algorithm for Netflix Originals IMDb Score Prediction

Objective:

This algorithm aims to guide the development of a predictive model for IMDb scores of Netflix Originals using the provided dataset. It covers essential steps, including feature engineering, model training, and evaluation, to ensure accurate predictions.

Steps:

1. Load and Preprocess the Netflix Originals Dataset:

Load the dataset, which includes information on Netflix Original films, such as title, genre, premiere date, runtime, IMDb scores, and available languages.

Ensure that you understand the dataset's structure and contents.

2. Feature Engineering:

Review the dataset to identify which features will be used for predicting IMDb scores. In this case, "Genre," "Runtime," and "Language" are potential features.

Handle any missing data. It appears that the dataset does not have any missing values.

Encode categorical data, such as "Language," using techniques like label encoding or one-hot encoding to convert them into a numerical format.

3. Feature Scaling

Analyze the dataset and determine if feature scaling is required. Some machine learning algorithms benefit from scaled features.

If needed, apply feature scaling to numerical features. For example, you can use standardization to scale the "Runtime" feature.

4. Split the Dataset:

Split the dataset into training and testing sets to assess the model's performance.

A common split ratio is 80% for training and 20% for testing. Ensure that the split is random to avoid any potential biases.

5. Select a Machine Learning Model:

-Choose an appropriate machine learning model for regression tasks.

6. Train the Model:

Initialize the chosen model.

Fit the model to the training data, using the selected features (e.g., "Genre," "Runtime," and "Language") as input and IMDb scores as the target variable.

During training, the model will learn patterns in the data.

7. Make Predictions:

Utilize the trained model to make IMDb score predictions on the testing data.
The model predicts IMDb scores based on the test feature data.

8. Evaluate the Model:

Assess the model's performance using regression evaluation metrics. Common metrics include:

Mean Absolute Error (MAE): Measures the average absolute difference between predicted and actual IMDb scores.

Mean Squared Error (MSE): Measures the average of the squared differences between predicted and actual IMDb scores.

Root Mean Squared Error (RMSE): The square root of MSE, providing error in the original IMDb score units.

R-squared (R2): Measures the proportion of the variance in IMDb scores explained by the model.

Visualize the results, such as scatter plots comparing actual IMDb scores vs. predicted IMDb scores or distribution plots.

This algorithm provides a structured approach to developing a IMDb score prediction model specifically tailored to the Netflix Originals dataset.

Execution of the model:

Importing the necessary libraries:

In [28]:
```
# 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
```

Train test split:

In [29]:
```
# 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)
```

Linear Model:

In [30]:
```
# 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)
```

Random Forest:

In [44]:
```
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
```

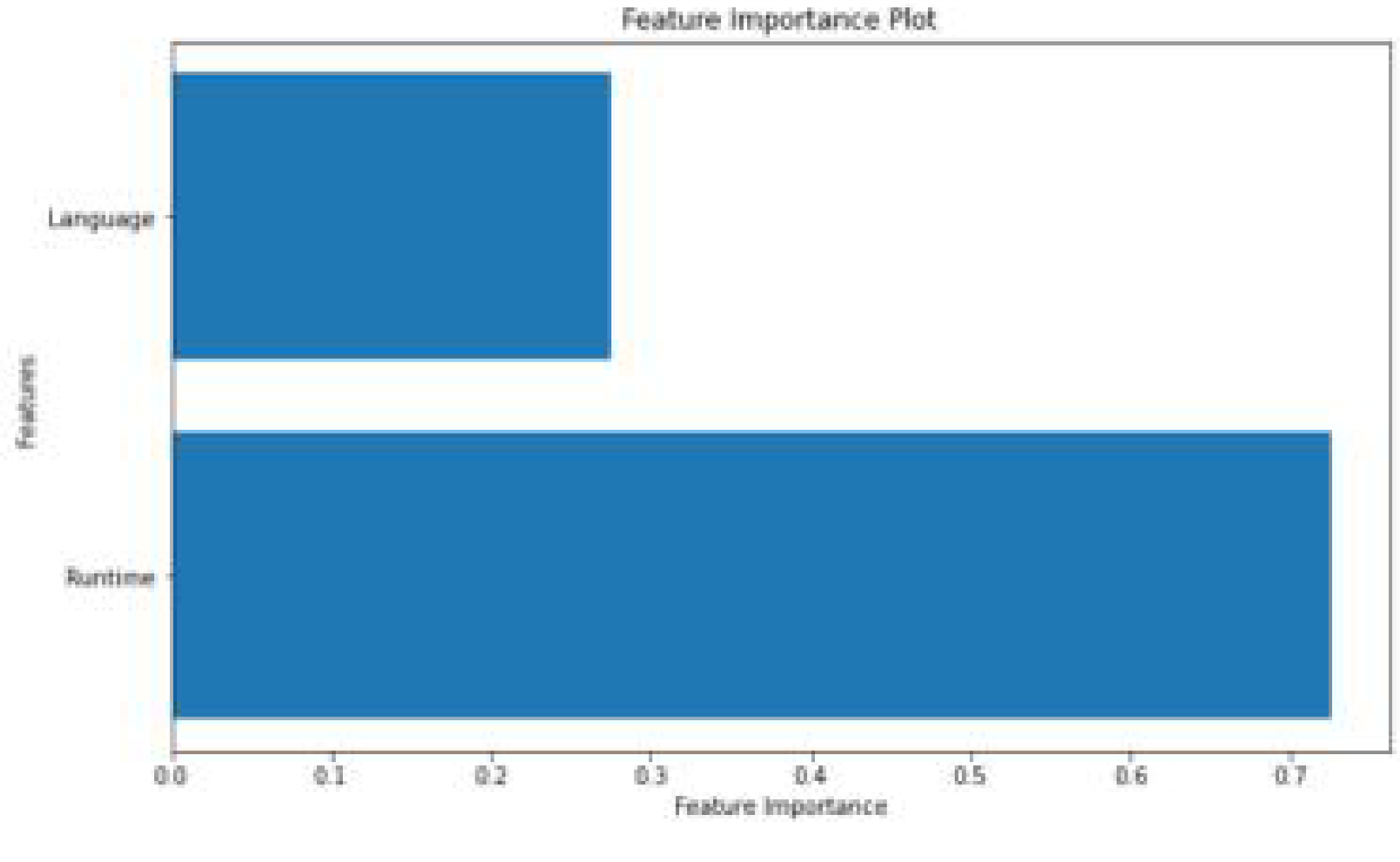
In [45]:
```
X = df.drop(["IMDB Score", "Title", "Genre", "Premiere"], axis=1)
y = df["IMDB Score"]

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

In [46]:
```
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)
```

Out[46]:
```
RandomForestRegressor(random_state=42)
```

Feature importance plot for Random Forest

In [47]:
```
if isinstance(model, RandomForestRegressor):
    feature_importance = model.feature_importances_
    feature_names = X_train.columns
    plt.figure(figsize=(10, 6))
    plt.barh(feature_names, feature_importance)
    plt.xlabel("Feature Importance")
    plt.ylabel("Features")
    plt.title("Feature Importance Plot")
    plt.show()
```



Evaluating the Model:

Using MAE, MSE, RMSE and R2

In [31]:
```
# 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
```

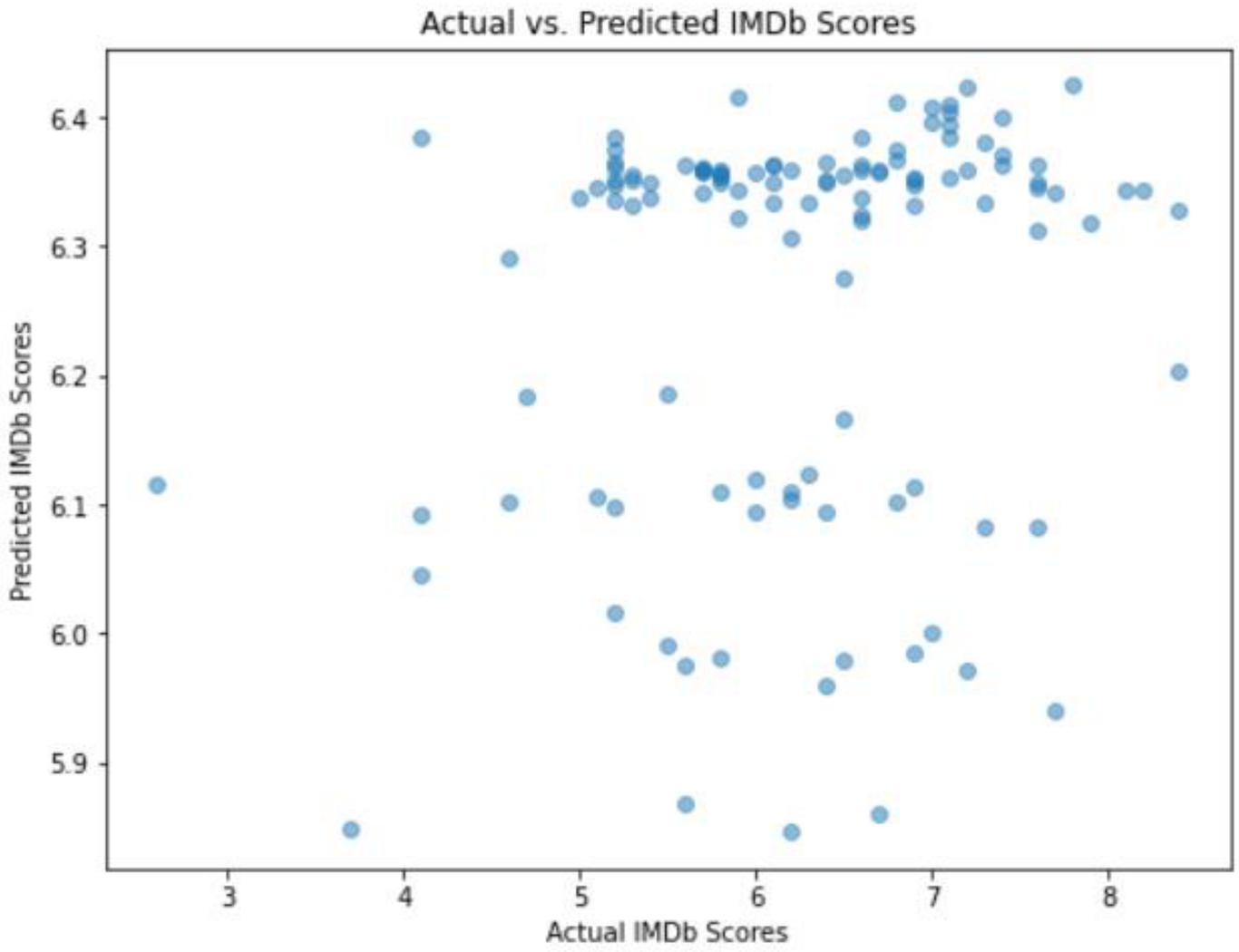
Visualization of the result:

Importing the libraries:

```
In [32]: import matplotlib.pyplot as plt
         import seaborn as sns
```

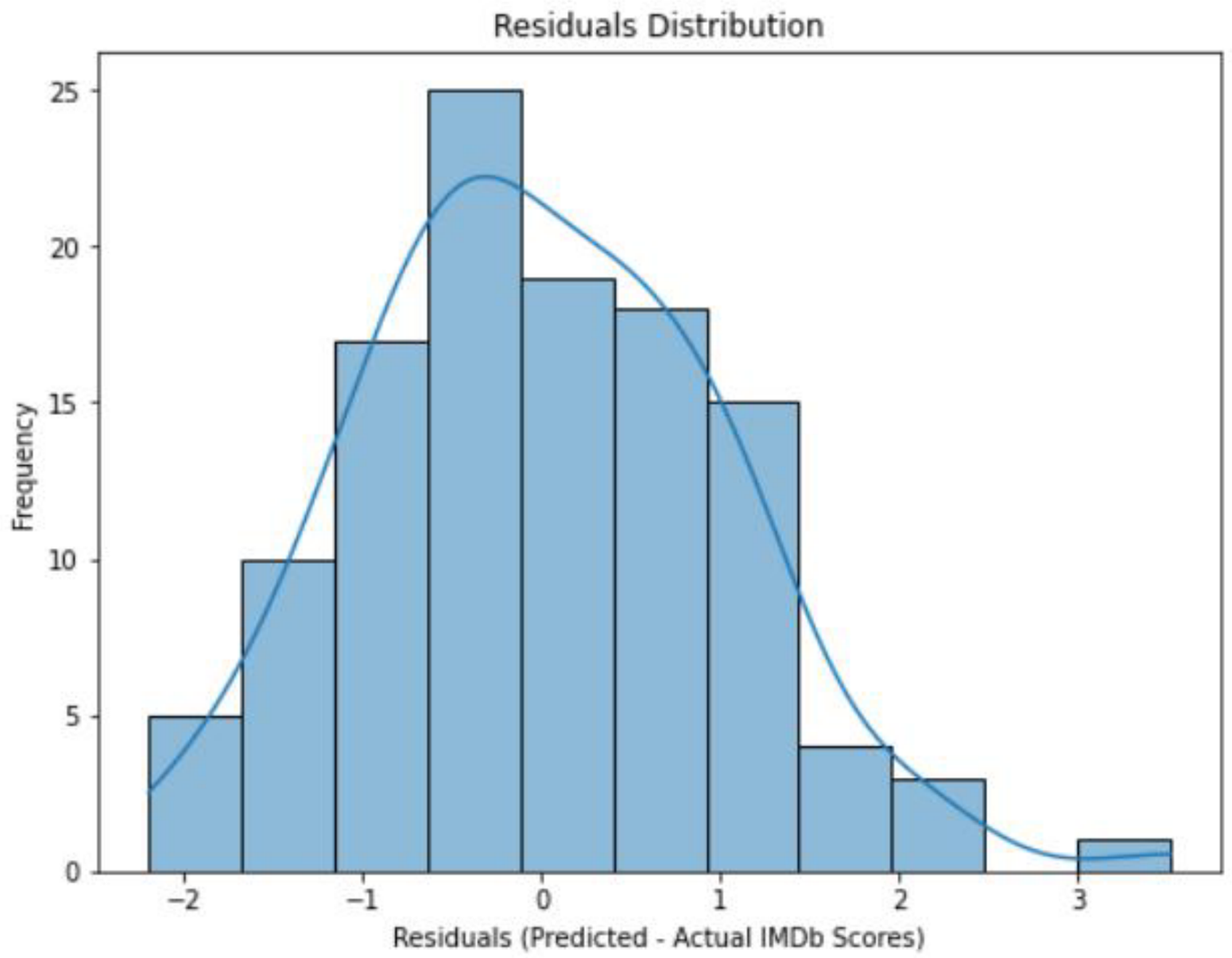
Actual vs, Predicted IMDB Scores

```
In [33]: # Scatter plot of actual IMDb scores vs. predicted IMDb scores
         plt.figure(figsize=(8, 6))
         plt.scatter(y_test, y_pred, alpha=0.5)
         plt.xlabel("Actual IMDb Scores")
         plt.ylabel("Predicted IMDb Scores")
         plt.title("Actual vs. Predicted IMDb Scores")
         plt.show()
```



Residual plot:

In [34]:
```
# 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()
```



In this phase, we embarked on the journey of building an IMDb score prediction model for Netflix original films. We began by loading and preprocessing the dataset, which included handling missing data, encoding categorical features, and scaling numerical attributes.

Our model selection led us to a Random Forest Regressor, which has the advantage of capturing complex relationships within the data. After training the model, we evaluated its performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and the R-squared (R2) coefficient. These metrics allowed us to assess the accuracy of our predictions.

Visualizations, including feature importance plots, residual plot and scatter plot provided additional insights into the model's performance. This comprehensive process equipped us with a powerful tool for predicting IMDb scores, which can be invaluable for filmmakers, content creators, and movie enthusiasts in assessing the potential success of Netflix original films.