**Applied DataScience :**

**Project Name: Predicting IMDb Scores**

Project Description: Develop a machine learning model to predict the IMDb scores of movies available on Films based on their genre, premiere date, runtime, and language. The model aims to accurately estimate the popularity of movies to assist users in discovering highly rated films that align with their preferences.

Phase 3: Development Part 1

Description :

Begin building the IMDb score prediction model by loading and preprocessing the dataset.

Load the movie dataset and preprocess the data for analysis.

Dataset Link: https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores

Working Procedure :

To load and preprocess the Netflix Originals IMDb Scores dataset from Kaggle, we can use the following steps:

Step 1:

Install the necessary Python libraries

Step 2 :

Load the dataset

# Load the dataset from the Kaggle website

Step 3 :

Explore the dataset

# Print the first 5 rows of the dataset

# Print the basic information about the dataset

Step 4 :

Preprocess the data

Handle missing values: There are no missing values in the dataset.

Convert categorical features to numerical features:

# Define a function to convert categorical features to numerical features

# Encode the Genre feature

# Encode the Language feature

Step 5 :

Scale the numerical features

# Define a function to scale numerical features

Step 6 :

Split the dataset into training and test sets.

Conclusion:

We have now loaded and preprocessed the Netflix Originals IMDb Scores dataset for analysis. The next step is to build a machine learning model to predict IMDb scores.

Program for an above steps :

In[1] : import pandas as pd

# Load the dataset from the Kaggle website

In [2] : netflix\_originals = pd.read\_csv(‘https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores/download’)

Out[2] :

Title Year Genre Language Runtime IMDB Score

0 Bird Box 2018 Thriller English 124 7.1

1 Roma 2018 Drama Spanish 135 7.7

2 6 Underground 2019 Action English 128 6.1

3 The Irishmen 2019 Crime English 209 7.9

4 Triple Frontier 2019 Action English 125 6.4

… … … … … … …

100 The Old Guard 2020 Action English 125 6.7

101 The Mitchells vs. the Machines 2021 Animation English 112 7.7

102 Don’t Look Up 2021 Comedy-Drama English 145 7.3

103 The Princess Switch 3: Romancing the Star 2021 Comedy English 105 5.7

104 The Adam Project 2022 Sci-Fi English 106 6.7

[105 rows x 6 columns]

# Print the first 5 rows of the dataset

In [3] : netflix\_originals.head()

Out [3] :

Title Year Genre Language Runtime IMDB Score

0 Bird Box 2018 Thriller English 124 7.1

1 Roma 2018 Drama Spanish 135 7.7

2 6 Underground 2019 Action English 128 6.1

3 The Irishmen 2019 Crime English 209 7.9

4 Triple Frontier 2019 Action English 125 6.4

<class ‘pandas.core.frame.DataFrame’>

RangeIndex: 105 entries, 0 to 104

# Print the basic information about the dataset

In [4] : netflix\_originals.info()

Out [4] :

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Title 105 non-null object

1 Year 105 non-null int64

2 Genre 105 non-null object

3 Language 105 non-null object

4 Runtime 105 non-null int64

5 IMDB Score 105 non-null float64

Dtypes: float64(1), int64(2), object(3)

Memory usage: 5.0+ KB

# Check for missing values

In [5] : netflix\_originals.isnull().sum()

Out [5] :

Title 0

Year 0

Genre 0

Language 0

Runtime 0

IMDB Score 0

Dtype: int64

# Define a function to convert categorical features to numerical features

In [6] : def encode\_categorical\_feature(df, column):

Return pd.get\_dummies(df[column], drop\_first=True)

# Encode the Genre feature

Netflix\_originals = encode\_categorical\_feature(netflix\_originals, ‘Genre’)

# Encode the Language feature

Netflix\_originals = encode\_categorical\_feature(netflix\_originals, ‘Language’)

Out [6] :

Title Year Documentary Drama Sci-Fi Thriller Language Runtime IMDB Score

0 Bird Box 2018 0.0

# Define a function to scale numerical features

In [7] : from sklearn.preprocessing import StandardScaler

Def scale\_numerical\_features(df, columns):

Scaler = StandardScaler()

Scaled\_df = scaler.fit\_transform(df[columns])

Return scaled\_df

# Scale the numerical features

Numerical\_features = [‘Runtime’]

Netflix\_originals = pd.concat([netflix\_originals, scale\_numerical\_features(netflix\_originals, numerical\_features)], axis=1)

Out [7] :

Title Year Documentary Drama Sci-Fi Thriller Language Runtime IMDB Score Runtime\_scaled

0 Bird Box 2018 0.0 X 0.973059 0.121793 0.0 0.0 1.0 English 124.0 7.1 1.213430

1 Roma 2018 0.0 X 1.193405 0.0 0.0 0.0 Spanish 135.0 7.7 1.365571

2 6 Underground 2019 0.0 X 0.0 0.0 1.0 0.0 English 128.0 6.1 1.267511

3 The Irishmen 2019 0.0 X 0.0 0.0 0.0 0.0 English 209.0 7.9 2.087592

4 Triple Frontier 2019 0.0 X 0.0 0.0 1.0 0.0 English 125.0 6.4 1.241561

… … … … … … … … … … …

100 The Old Guard 2020 0.0 X 0.0 1.0 0.0 0.0 English 125.0 6.7 1.241561

101 The Mitchells vs. the Machines 2021 0.0 X 1.0 0.0 0.0 0.0 English 112.0 7.7 1.121793

102 Don’t Look Up 2021 0.0 X 0.0 0.0 0.0 1.0 English 145.0 7.3 1.429630

103 The Princess Switch 3: Romancing the Star 2021 0.0 X 0.0 0.0 0.0 0.0 English 105.0 5.7 1.043821

104 The Adam Project 2022 0.0 X 0.0 0.0 0.0 0.0 English 106.0 6.7 1.061842

[105 rows x 10 columns]

In [8] : from sklearn.model\_selection import train\_test\_split

# Split the dataset into training and test sets

X = netflix\_originals.drop(‘IMDB Score’, axis=1)

Y = netflix\_originals[‘IMDB Score’]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42)

Out [8] :

(78, 27)

This means that the training set contains 78 samples and the test set contains 27 samples.