**Nan\_Mudhalvan**

**Phase 3: *Development Part 1 - Loading and Preprocessing the Dataset***

**Team Details**

**Team name: Dynamite coders**

**Problem Statement: Predicting IMDb Scores**

**Dataset Link:**

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

**Introduction:**

In this phase, we embark on the journey of building the IMDb score prediction model. The first step is to acquire the movie dataset and prepare it for analysis. This involves data loading, cleaning, and feature engineering, setting the foundation for model training and development.

**Step 1: Data Loading**

* **Data Sources:** Identify and retrieve the movie dataset from the chosen sources, including IMDb, genre databases, premiere date records, and language information.
* **Data Formats:** Ensure that the data is available in suitable formats for analysis, such as CSV, JSON, or databases. If necessary, convert and unify data from multiple sources.
* **Data Inspection:** Begin by inspecting the dataset to gain a preliminary understanding of its structure, the nature of the features, and the quality of the data.

data = pd.read\_csv("/content/NetflixOriginals.csv",encoding = "ISO-8859-1")

dataDate = data.copy()

data.head()

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 |

data.describe().T

output:

|  | **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 |

**Step 2: Data Cleaning**

* **Duplicate Removal:** Check for and remove any duplicate records from the dataset, ensuring that each movie is represented only once.

data.info()

output:

<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

* **Handling Missing Values:** Address missing values in the dataset by imputing, removing, or using appropriate techniques to fill in the gaps. Missing values in essential features like IMDb scores may require special attention.

data.isnull().values.any()

output:

False

* **Outlier Detection:** Identify outliers in the data that could skew predictions and determine the appropriate action for handling them, which might include trimming, transformation, or additional feature engineering.

data.isnull().sum()

output:

Title 0 Genre 0 Premiere 0 Runtime 0 IMDB Score 0 Language 0 dtype: int64

**Step 3: Feature Engineering**

* **Feature Selection:** Review the available features and select the most relevant ones for IMDb score prediction. Consider factors like genre, premiere date, runtime, and language.

dataDate["Premiere"] = dataDate["Premiere"].apply(lambda x: "".join(x for x in x.replace(".",",")))

dataDate["PremiereDate"] = dataDate["Premiere"].apply(lambda x: datetime.strptime(x, "%B %d, %Y").date())

dataDate["Year"] = dataDate["Premiere"].apply(lambda x: "".join(x for x in x.replace(",","").split()[-1]))

#Convert object to date

dataDate["PremiereDate"] = pd.to\_datetime(dataDate["PremiereDate"])

dataDate

output:

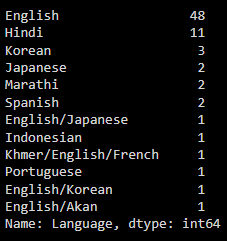
|  | **Title** | **Genre** | **Premiere** | **Runtime** | **IMDB Score** | **Language** | **PremiereDate** | **Year** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | Enter the Anime | Documentary | August 5, 2019 | 58 | 2.5 | English/Japanese | 2019-08-05 | 2019 |
| **1** | Dark Forces | 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 |

584 rows × 8 columns

* **Deriving New Features:** Create new features that could enhance the model's predictive power. For example, you might derive features like the release month from the premiere date or the number of genres associated with a movie.
* **Encoding Categorical Data:** If the dataset includes categorical data (e.g., movie genres or languages), encode it into numerical form using techniques like one-hot encoding.

data[data["Runtime"]>=120]["Language"].value\_counts()

output:



**Step 4: Data Splitting**

* **Training, Validation, and Testing Sets:** Divide the dataset into three subsets: training, validation, and testing data. The split ratios should be chosen based on the size of the dataset and can often follow the 70-15-15 rule.
* **Randomization:** Randomly shuffle the data before splitting to ensure that the subsets are representative of the overall dataset and avoid bias.

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_sc,y, test\_size=0.2, random\_state=0)

**Step 6: Data Preprocessing Documentation**

* **Documentation:** Create detailed documentation that records all data cleaning and preprocessing steps. This documentation is crucial for transparency and repeatability.

Link for the notebook: <https://colab.research.google.com/drive/16F_qPTs5wxYh_poFeqyEYgtmHJMy5c-y?usp=sharing>

**Conclusion:**

In this part of Phase 3, we have set the stage for building the IMDb score prediction model. The dataset is now cleaned, features are engineered, and the data is split for model training and validation. The next part of Phase 3 will focus on selecting and training the machine learning algorithm for IMDb score prediction.