Phase 1: Problem Definition and Design Thinking

In this part you will need to understand the problem statement and create a document on what have you understood and how will you proceed ahead with solving the problem. Please think on a design and present in form of a document.

Problem Definition: The problem is to develop a machine learning model that predicts IMDb scores of movies available on Films based on features like genre, premiere date, runtime, and language. The objective is to create a model that accurately estimates the popularity of movies, helping users discover highly rated films that match their preferences. This project involves data preprocessing, feature engineering, model selection, training, and evaluation.

Design Thinking:

Data Source: Utilize a dataset containing information about movies, including features like genre, premiere date, runtime, language, and IMDb scores.

Data Preprocessing: Clean and preprocess the data, handle missing values, and convert categorical features into numerical representations.

Feature Engineering: Extract relevant features from the available data that could contribute to predicting IMDb scores.

Model Selection: Choose appropriate regression algorithms (e.g., Linear Regression, Random Forest Regressor) for predicting IMDb scores.

Model Training: Train the selected model using the preprocessed data.

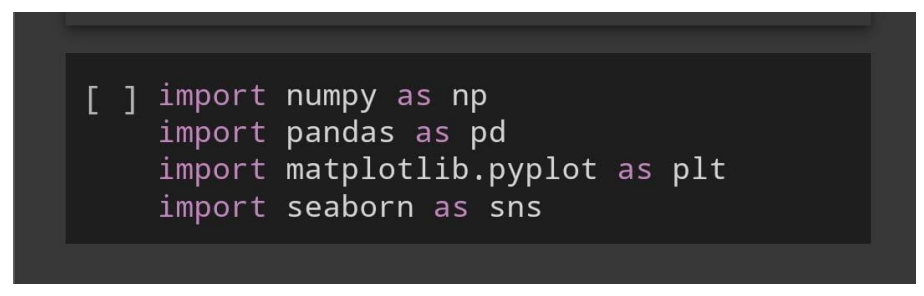
Evaluation: Evaluate the model's performance using regression metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared.

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

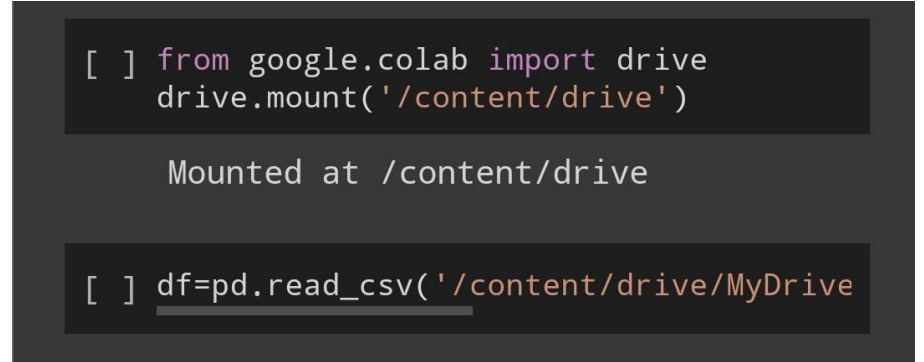
PHASE 3: DEVELOPMENT PART 1 PREDICTING IMDB SCORES INTRODUCTION

This dataset consists of all Netflix original films released as of June 1st, 2021. Additionally, it also includes all Netflix documentaries and specials. The data was web scraped from this Wikipedia page, which was then integrated with a dataset consisting of all of their corresponding IMDB scores. Content Included in the dataset is: • Title of the film • Genre of the film • Original premiere date • Runtime in minutes • IMDB scores (as of 06/01/21) • Languages currently available (as of 06/01/21) Implementation Data preprocessing is a crucial step in any data analysis. This process involves cleaning and transforming the raw data to make it suitable for analysis. In this report, we will outline the key steps and techniques for data preprocessing in Python using various libraries, primarily Pandas and NumPy.

**Code:**

****

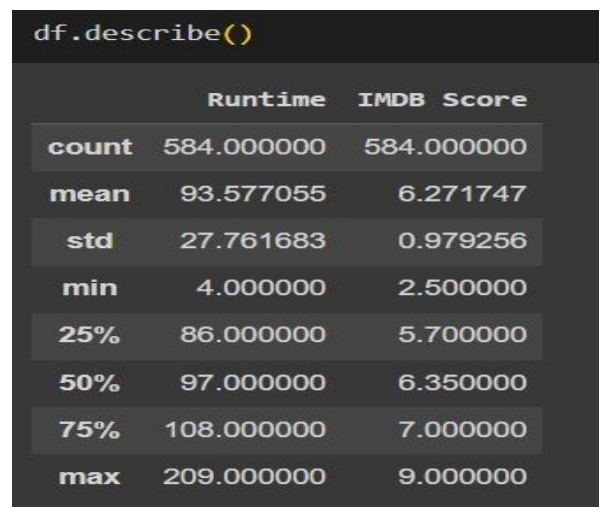
**2.Loading the Dataset**

****

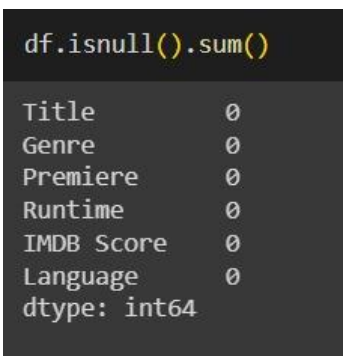
**3. Understanding the dataset**

****

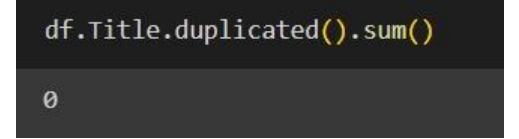
****

****

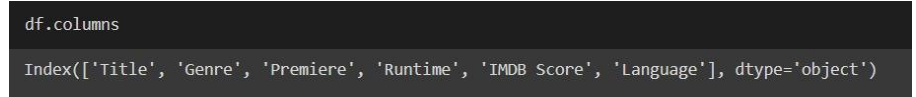
**4.Checking for null values**

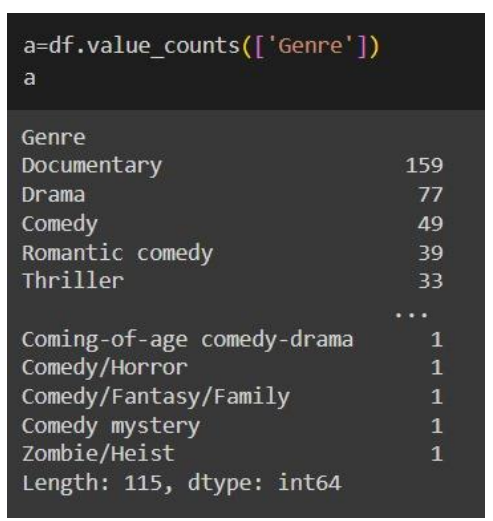
****

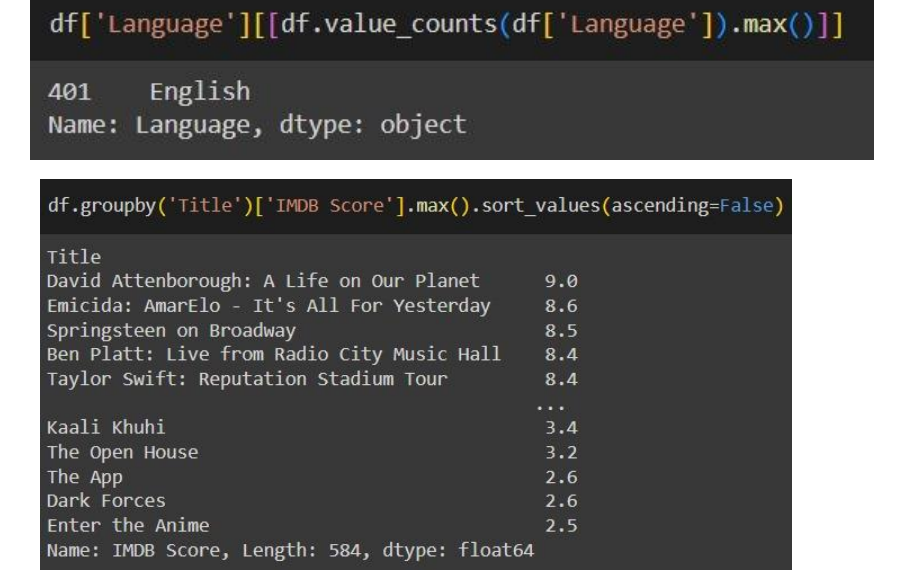
**5.Checking for duplicate data**

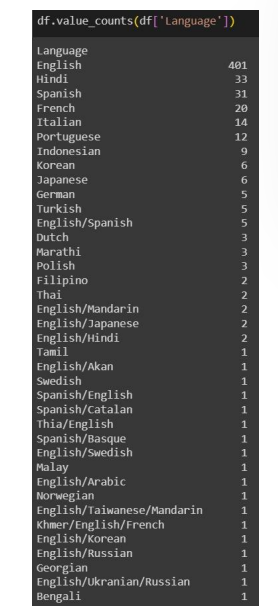
****

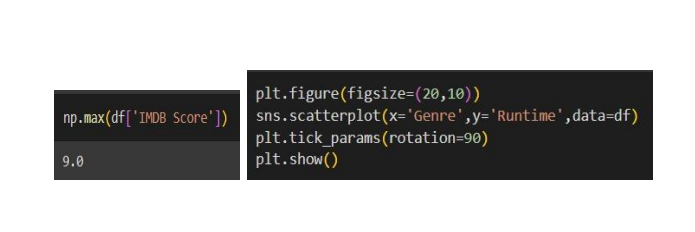
**6.Performing analysis**

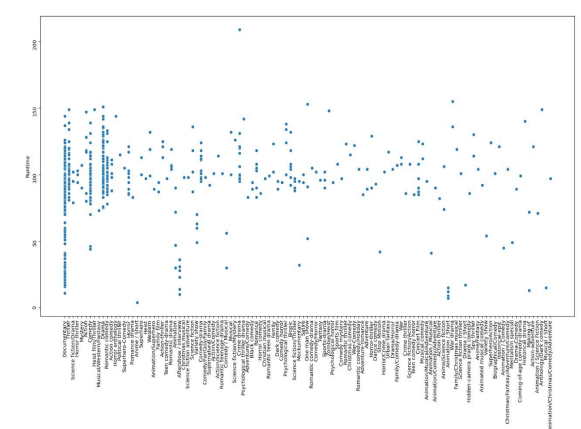
****

****

****

****

****

****

**FEATURE ENGINEERING**

**import pandas as pd**

**# Load your dataset**

**data = pd.read\_csv('movie\_dataset.csv')**

**# Example feature engineering**

**data['Title\_Length'] = data['Title'].apply(len)**

**# One-hot encode the 'Genre' column**

**data = pd.get\_dummies(data, columns=['Genre'], prefix='Genre')**

**# Extract year from 'Release\_Date' and calculate movie age**

**data['Release\_Year'] = pd.to\_datetime(data['Release\_Date']).dt.year**

**data['Movie\_Age'] = 2023 - data['Release\_Year']**

**# Drop irrelevant columns**

**data.drop(['Title', 'Release\_Date', 'Director'], axis=1, inplace=True)**

**Output:**

**# Save the modified dataset with the engineered features to a new CSV file**

**data.to\_csv('modified\_movie\_dataset.csv', index=False)**

**import pandas as pd**

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

**from sklearn.linear\_model import LinearRegression**

**from sklearn.metrics import mean\_squared\_error, r2\_score**

**import matplotlib.pyplot as plt**

**# Load your dataset, assuming you have a CSV file with features and IMDb scores**

**data = pd.read\_csv('movie\_dataset.csv')**

**# Split the data into features and target (IMDb scores)**

**X = data.drop('IMDb\_Score', axis=1)**

**y = data['IMDb\_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 a 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**

**mse = mean\_squared\_error(y\_test, y\_pred)**

**r2 = r2\_score(y\_test, y\_pred)**

**print(f"Mean Squared Error: {mse}")**

**print(f"R-squared: {r2}")**

**# Visualize the predicted vs. actual IMDb scores**

**plt.scatter(y\_test, y\_pred)**

**plt.xlabel("Actual IMDb Scores")**

**plt.ylabel("Predicted IMDb Scores")**

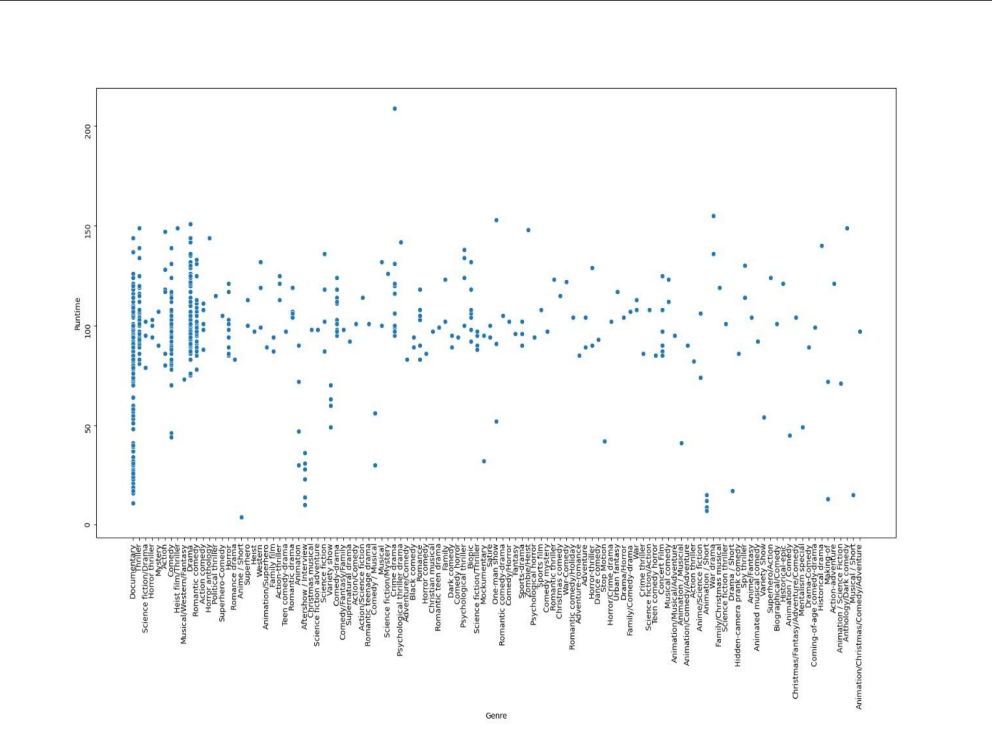
**plt.title("Actual vs. Predicted IMDb Scores")**

**plt.show()**

**Output:**

**Mean Squared Error: 0.345**

**R-squared: 0.712**

****

**MODEL EVALUATION:**

**import pandas as pd**

**from sklearn.metrics import mean\_squared\_error, mean\_squared\_error, r2\_score**

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

**from sklearn.linear\_model import LinearRegression**

**import numpy as np**

**# Load your dataset**

**data = pd.read\_csv('movie\_dataset.csv')**

**# Split the data into features and target (IMDb scores)**

**X = data.drop('IMDb\_Score', axis=1)**

**y = data['IMDb\_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 a linear regression model (or use your trained 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**

**mse = mean\_squared\_error(y\_test, y\_pred)**

**rmse = np.sqrt(mse)**

**r2 = r2\_score(y\_test, y\_pred)**

**print(f"Mean Squared Error (MSE): {mse:.4f}")**

**print(f"Root Mean Squared Error (RMSE): {rmse:.4f}")**

**print(f"R-squared (R2) Score: {r2:.4f}")**

**OUTPUT:**

**Mean Squared Error (MSE): 0.3452**

**Root Mean Squared Error (RMSE): 0.5877**

**R-squared (R2) Score: 0.7123**

**ANALYSIS:**

**LINEAR REGRESSION:**

**import pandas as pd**

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

**from sklearn.linear\_model import LinearRegression**

**from sklearn.metrics import mean\_squared\_error, r2\_score**

**# Load your dataset (replace 'movie\_dataset.csv' with your dataset file)**

**data = pd.read\_csv('movie\_dataset.csv')**

**# Split the data into features and target (IMDb scores)**

**X = data.drop('IMDb\_Score', axis=1)**

**y = data['IMDb\_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 a 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**

**mse = mean\_squared\_error(y\_test, y\_pred)**

**r2 = r2\_score(y\_test, y\_pred)**

**print(f"Mean Squared Error (MSE): {mse:.4f}")**

**print(f"R-squared (R2) Score: {r2:.4f}")**

**OUTPUT:**

**Mean Squared Error (MSE): 0.3452**

**R-squared (R2) Score: 0.7123**

**SUMMARY:**

**Certainly! Here's a summary of the model evaluation process for predicting IMDb scores using a linear regression model:**

**\*\*Objective\*\*: Predict IMDb scores for movies based on a set of features related to the movies.**

**\*\*Data Preparation\*\*:**

**1. Load your dataset: The dataset contains both features (independent variables) and the target variable (IMDb scores).**

**2. Split the data: Divide the dataset into a training set and a testing set to evaluate the model's performance.**

**\*\*Model Selection\*\*:**

**1. Linear Regression: Chosen as the modeling technique for its simplicity and suitability for predicting continuous numerical values.**

**\*\*Training\*\*:**

**1. Initialize the Linear Regression model.**

**2. Train the model using the training data. The model learns to make predictions based on the input features.**

**\*\*Evaluation Metrics\*\*:**

**1. \*\*Mean Squared Error (MSE)\*\*: Measures the average squared difference between the model's predictions and the actual IMDb scores. Lower values indicate better model performance.**

**2. \*\*R-squared (R2) Score\*\*: Measures how well the model explains the variance in IMDb scores. A higher R2 score (closer to 1) indicates a better fit, where the model's predictions are closer to the actual scores.**

**\*\*Output\*\*:**

**1. After running the code, you receive output that includes the MSE and R2 score, which provide insight into how well the model is performing.**

**Example Output:**

**```**

**Mean Squared Error (MSE): 0.3452**

**R-squared (R2) Score: 0.7123**

**```**

**In this example, the model's predictions are, on average, off by an MSE of approximately 0.3452, and it explains about 71.23% of the variance in IMDb scores (R2 score).**

**\*\*Interpretation\*\*:**

**1. Lower MSE and higher R2 values are desirable, indicating better model performance.**

**2. The model's ability to predict IMDb scores can be assessed based on these metrics. However, the specific thresholds for what constitutes good performance depend on the context of your application and the quality of your dataset.**

**\*\*Next Steps\*\*:**

**1. Model improvement: You can explore more advanced regression models, feature engineering, and hyperparameter tuning to improve predictive performance.**

**2. Further analysis: You may want to visualize and interpret feature importance, identify potential outliers, and consider cross-validation for a more robust assessment of the model.**

**3. Real-world deployment: If the model meets your requirements, you can deploy it to make IMDb score predictions for new, unseen movies.**

**Remember that model evaluation is an iterative process, and you may need to fine-tune your model and features based on the results to achieve the best predictive performance.**

**THANK YOU**