**PROJECT TITLE: PREDICTING IMDB SCORES**

**(PHASE 4)**

**Team Members:**

* *GEETIKA.R-715521106014*
* *MIRUTHULA.P.V-715521106029*
* *SRI KARTHICK RAJAN.N-715521106050*
* *THANUJ.R-715521106052*
* *THARUN SRIPAL.A.D-715521106053*
* *HARSHAN.B-715521106304*

**PROBLEM STATEMENT:**

The problem at hand is to develop a machine learning model that can accurately predict the IMDb scores of movies based on several key attributes, including genre, premiere date, runtime, and language. IMDb scores represent the perceived quality and popularity of movies, making this prediction task valuable for assisting users in discovering high-rated films that align with their preferences.

**DATASET:**

Data set link: <https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores/>

The data set includes details about the film's title, genre, Premiere, runtime (the length of the film), IMDb ratings, and language.

**Feature Encoding Or Engineering:**

**Exporting libraries:**

from sklearn.preprocessing import LabelEncoder, OneHotEncoder, StandardScaler

from sklearn.compose import ColumnTransformer

**Genreband language based Features:**

* Extracting genre , language from the ‘Genre’, ’language’ column, which is a common approach for the series data.

Source code:

categorical\_cols = ['Genre', 'Language']

**Applying One Code Encoding:**

# **Create a ColumnTransformer to apply one-hot encoding**

preprocessor = ColumnTransformer(

transformers=[

('cat', OneHotEncoder(), categorical\_cols),

],

remainder='passthrough' # Include other columns as-is

)

# **Fit and transform the training data**

X\_train = preprocessor.fit\_transform(X\_train)

# **Transform the test data using the same preprocessor**

X\_test = preprocessor.transform(X\_test)

**Modal Training And Evaluation:**

* Evaluating the model's performance on the testing data. Common evaluation metrics include accuracy, precision, recall, F1-score, and ROC AUC, depending on the nature of your problem.
* Use our trained model to make predictions on your dataset. If we have a test set, we use that for prediction.

Depending on the nature of our problem, we should choose appropriate evaluation metrics. Common metrics for binary classification include:

* Accuracy: The proportion of correctly predicted cases.
* Precision: The ratio of true positive predictions to the total predicted positive cases.
* Recall: The ratio of true positive predictions to the total actual positive cases.
* F1-score: The harmonic mean of precision and recall.
* ROC AUC: The area under the Receiver Operating Characteristic curve.
* Interpreting the evaluation metrics and confusion matrix to understand how well our model is performing. Depending on the problem, we may need to fine-tune our model or consider additional feature engineering.

**Source code:**

# Define the target variable

y = ds\_date['IMDB Score']

# Split the data into training and testing

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# **Create a Random Forest Regressor model**

model = RandomForestRegressor(n\_estimators=100, random\_state=42)

# **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**

mae = mean\_absolute\_error(y\_test, y\_pred)

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

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

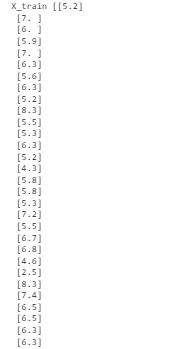
print(f"Mean Absolute Error (MAE): {mae}")

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

print(f"R-squared (R^2): {r2}")

**OUTPUT:**

**ENCODING**  **MODAL EVALUATION**

******