**Predicting IMDb Score**

**Date : 28/10/2023**

**Team : 3866**

**1. Problem Statement**

The central problem of this project is to create a system capable of predicting IMDb (Internet Movie Database) scores for movies, aiming to estimate the popularity and quality of films accurately. The model will be trained on a comprehensive dataset containing movie attributes such as genre, premiere date, runtime, and language, and it will provide IMDb score predictions as its output. The primary objective is to assist users in identifying highly rated movies that align with their preferences, enhancing their movie-watching experience. The project's success will be measured by the model's ability to deliver precise predictions, making it a valuable tool for both movie enthusiasts and film industry professionals. This predictive system seeks to streamline movie selection and recommendation processes by leveraging data-driven insights.

**2. Design and Innovation Strategies**

**2.1 - Data Collection**

Implement advanced data collection techniques, including web scraping to gather diverse movie-related data such as historical IMDb scores, movie metadata, user ratings, and reviews. Apply innovative feature engineering methods to extract meaningful insights from both structured and unstructured data sources. Techniques like sentiment analysis and keyword extraction will be explored to enhance prediction accuracy.

**2.2 - Model Selection and Training**

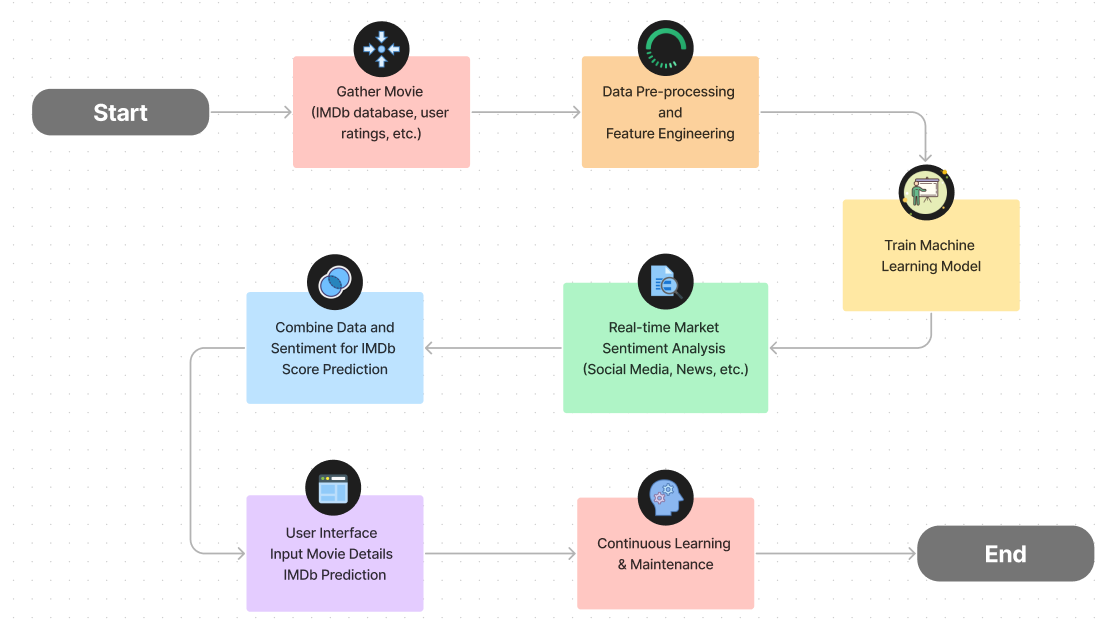
Explore a wide range of machine learning algorithms, including regression, decision trees, Naïve Bayes to determine the most effective model for IMDb score prediction. Investigate ensemble learning techniques, such as Random Forests and Gradient Boosting, to combine the strengths of multiple models and improve prediction accuracy.

**2.3 - Market Sentiment Analysis**

Integrate external data sources, such as real-time social media sentiment analysis and news sentiment, from the dataset to gauge market sentiment and its potential impact on IMDb scores.

**2.4 - Continuous Learning**

Establish a continuous learning framework that continuously updates the prediction model with real-time user feedback and new data. Implemented automated data pipelines for seamless data ingestion, processing, and model retraining, ensuring that the system remains up-to-date and accurate.



**3 - Dataset & Data collection:**

There are many datasets available online for movies, in our analysis we used a movie metadata form Kaggle.com to train our model.

Reference : <https://www.kaggle.com/code/saurav9786/imdb-score-prediction-for-movies/input>

This dataset contains the information about the movies. For a movie to be commercial success, it depends on various factors like director, actors, critic reviews and viewers reaction. IMDb score is one of the important factor to measure the movie's success. This dataset contains 28 columns. The dataset here gives the massive information about the movies and their IMDB scores respectively. We are going to analyse each and every factors which can influence the IMDb ratings so that we can predict better results. The movie with the higher IMDb score is more successful as compared to the movies with low IMDb score.

**4 – Choice Of Machine Learning Algorithm:**

**4.1. KNeighbour (K-Nearest Neighbours) - Accuracy: 70.04%:**

K-Nearest Neighbours is a distance-based algorithm, so data preprocessing is crucial. Standardizing or normalizing the data is important to ensure all features contribute equally to the distance calculation. KNN doesn't inherently perform feature extraction, but selecting relevant features during preprocessing can impact its performance.

**4.2. Decision Tree - Accuracy: 65.88%:**

Decision trees are not very sensitive to data scaling, but data cleaning and handling missing values are important. Outliers can affect tree structure. Decision trees can handle both categorical and numerical features, but they might not be efficient with high-dimensional data. Feature selection can help improve performance.

**4.3. Random Forest - Accuracy: 76.13%:**

Data Preprocessing: Random Forest, being an ensemble method of decision trees, benefits from similar preprocessing steps as Decision Trees. Data cleanliness and handling outliers are important. Feature Extraction: Random Forest can handle a wide range of features effectively, but again, feature selection can help improve its performance by reducing noise in the data.

**4.4. Naive Bayes Gaussian (NBG) - Accuracy: 29.27%:**

Naive Bayes methods are less sensitive to data preprocessing, but it's important to handle missing values and ensure that features are independent. Feature selection is important to ensure that only relevant features are used since Naive Bayes assumes feature independence.

**4.5. Naive Bayes Bernoulli (NBB) - Accuracy: 59.17%:**

Similar to NBG, Naive Bayes Bernoulli benefits from data preprocessing steps like handling missing values and ensuring that features are binary or binarized. Feature selection is essential, and you may need to perform binarization of features to fit the Bernoulli assumption.

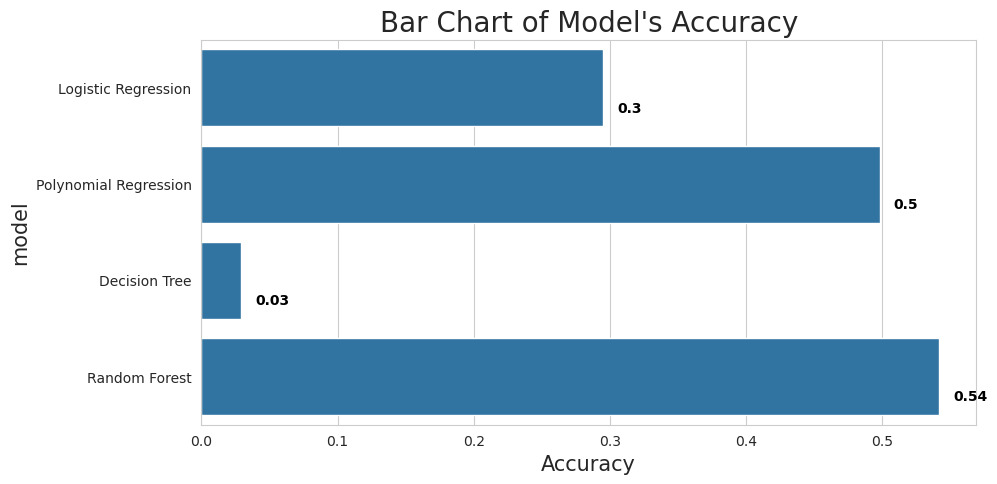
The differences in the performance of these algorithms may be attributed to several factors, including the quality of data preprocessing and feature extraction.

Random Forest performs the best because it can handle a variety of data types and often benefits from the inclusion of more relevant features, making it robust against noise.

K-Nearest Neighbours is also quite sensitive to data preprocessing and feature selection, so a well-preprocessed dataset with carefully selected features can lead to good performance.

Naive Bayes algorithms are more simplistic and may struggle with complex or high-dimensional data. Their performance can be significantly impacted by the choice of features and preprocessing steps.

Decision Tree's performance falls between Random Forest and KNeighbour. It can capture complex relationships but may overfit or underfit depending on the data and tree depth.



**5 – Conclusion :**

Based on the provided accuracy scores, if we consider accuracy as the sole criterion for choosing the best algorithm, then **Random Forest** with an accuracy of 76.13% has the highest accuracy score among the algorithms listed. Therefore, if accuracy is the primary metric of interest for your task, Random Forest would be the best choice among the options provided.