Predicting IMDb Score

Date: 31/10/2023

Team: 3866

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.

Introduction:

In the realm of cinema, predicting the IMDb scores of films is a project of paramount significance. Leveraging the power of regression analysis, this venture delves deep into the intricate relationships between diverse film attributes and IMDb ratings. It offers filmmakers and studios a valuable tool to enhance decision-making, marketing, and audience engagement. Through feature engineering, model selection, and rigorous evaluation metrics, we aim to unlock the potential and limitations of regression analysis for IMDb score prediction. This project seeks to empower industry professionals by revealing the art and science behind forecasting audience appreciation, ultimately reshaping the landscape of filmmaking.

Literature Survey:

1. **Use of Social Media Signals:** The use of social media signals such as Twitter and YouTube activity to predict IMDb movie ratings. These signals capture audience opinions and sentiments, providing a more accurate and timelier source of information for rating predictions.

2. **Important Movie Factors:** The significance of factors like the number of voted users, critics' reviews, Facebook likes, movie duration, and gross collection in influencing IMDb scores. Understanding these factors is crucial for accurate predictions.

3. **Machine Learning Algorithms:** The application of machine learning algorithms for predicting IMDb ratings. Random Forest is identified as the best-performing algorithm for accurate predictions, providing a valuable tool for movie success prediction.

4. **Text Classification and Deep Learning:** The text classification using deep learning techniques for IMDb score prediction. This approach considers user reviews and emotions, offering insights into the potential of advanced methods for predicting ratings.

5. **Data Quality and Trends:** The importance of data quality and the challenge of capturing changing audience preferences and market dynamics. It's crucial to ensure that the training data used for prediction is accurate and representative.

Design Thinking Process:

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.

Model Selection and Training:

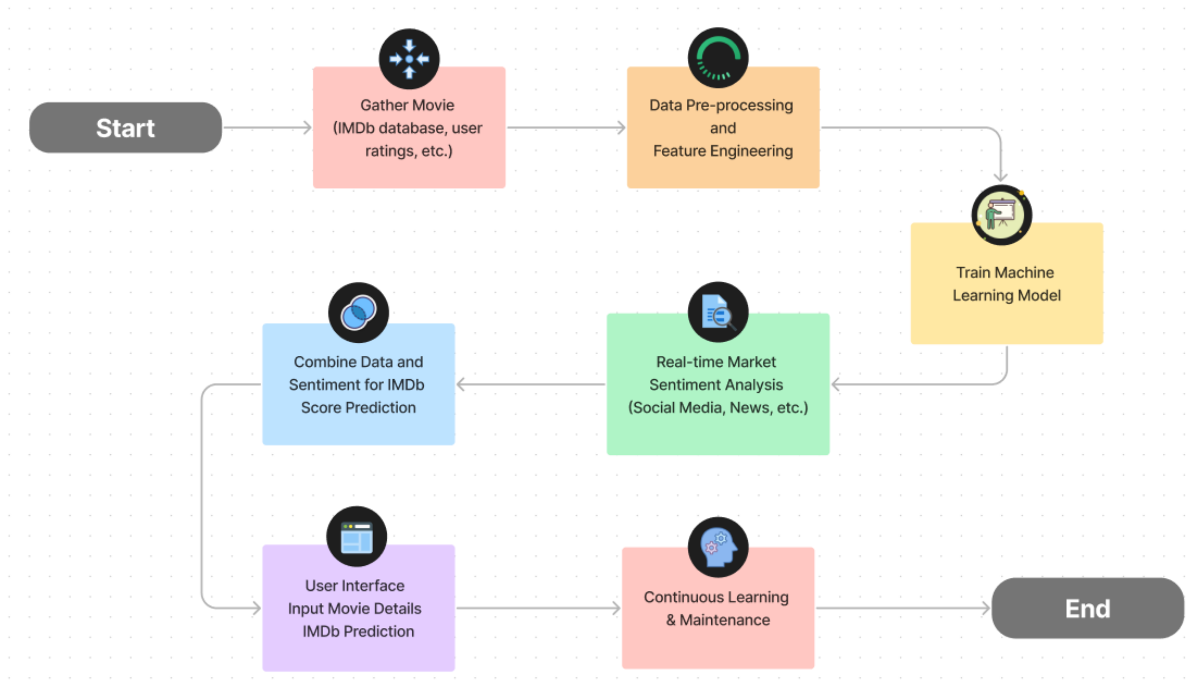
Explore a wide range of machine learning algorithms, including regression, decision trees, Naïve Bayes to determine the most effective model for IMDb score prediction. Investigate ensemble learning techniques, such as Random Forests and Gradient Boosting, to combine the strengths of multiple models and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

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.

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.


Start
Gather Movie (IMDb database, user ratings, etc.)
Data Pre-processing and Feature Engineering
Train Machine Learning Model
Real-time Market Sentiment Analysis (Social Media, News, etc.)
Combine Data and Sentiment for IMDb Score Prediction
User Interface Input Movie Details IMDb Prediction
Continuous Learning & Maintenance
End

Phases of development:

Phase 1: Importing Dependencies

This phase involves importing necessary Python libraries and modules. These libraries are required for data processing, visualization, and various machine learning tasks.

```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import OrdinalEncoder, StandardScaler, PolynomialFeatures
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, accuracy_score, r2_score, mean_absolute_error
from sklearn import metrics
from scipy import stats
```

Phase 2: Reading and Viewing Data

In this phase, the code reads a dataset from a CSV file named 'movie_metadata.csv' using Pandas.

```python
data = pd.read_csv('movie_metadata.csv')
```

Phase 3: Data Preprocessing and Exploration

Here, data preprocessing and exploration occur, including column selection, creating a new column for IMDb score categories, and encoding the 'genre' column. Outliers are also removed.

```python
df = df[['duration','director_facebook_likes','gross','budget','actor_1_facebook_likes','genres','imdb_score','num_critic_for_reviews','facenumber_in_poster']]
df['imdb_score_category'] = pd.cut(df.imdb_score, bins=[0, 4, 6, 8, 10], right=True, labels=['bad', 'average', 'good', 'excellent'])
df['other_actors_likes'] = df['actor_3_facebook_likes'] + df['actor_2_facebook_likes']
df['genre'] = df['genres'].str.split('|').str[0]
ord_enc = OrdinalEncoder()
df['genre_code'] = ord_enc.fit_transform(df[['genre']])
df.drop(df[(np.abs(stats.zscore(df['gross']))<3)],axis=1)
df.dropna(inplace=True)
```

Phase 4: Model Training

This phase includes data scaling, splitting the dataset into training and testing sets, and preparing the features and target values for machine learning models.

```python
sc = StandardScaler()
sc.fit(df[['duration','director_facebook_likes','actor_1_facebook_likes','other_actors_likes','budget','genre_code','num_voted_users','num_critic_for_reviews']])

X = np.array(sc.transform(df[['duration','director_facebook_likes','actor_1_facebook_likes','other_actors_likes','budget','genre_code','num_voted_users']]))
Y = np.array(df['imdb_score']).ravel()

xtrain, xtest, ytrain, ytest = train_test_split(X, Y, test_size=0.25, random_state=0)
```

Phase 5: Model Selection and Evaluation

In this final phase, machine learning models are trained and evaluated using R-squared scores (R2) to determine their accuracy. A bar chart is generated to visualize the accuracy of different models. The best model has been selected for further use.

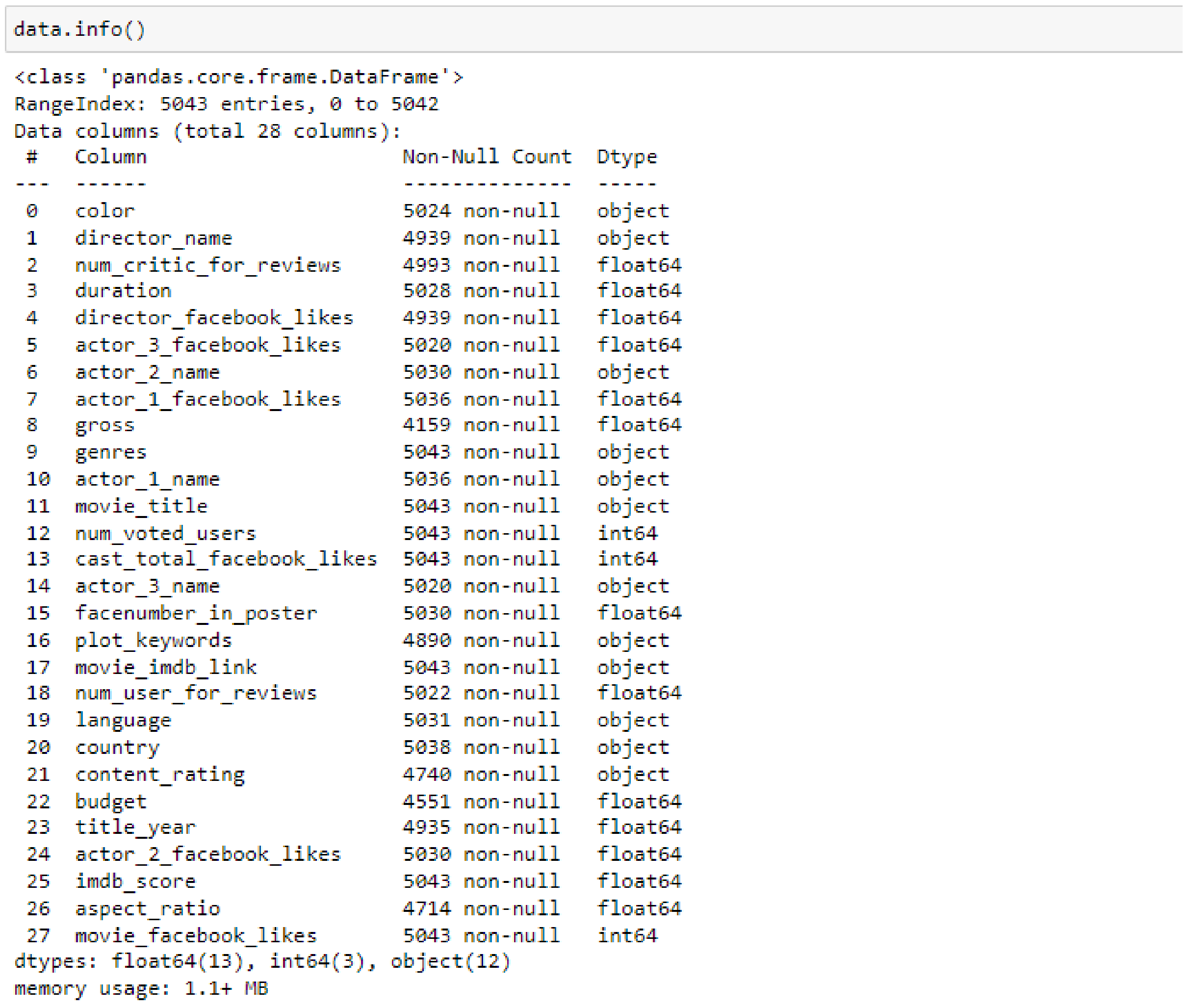
```
linreg = LinearRegression()
linreg.fit(xtrain, ytrain)
pred = linreg.predict(xtest)
LR = r2_score(ytest, pred)

polyreg = PolynomialFeatures(degree=3)
x_poly = polyreg.fit_transform(X)
linreg.fit(x_poly, Y)
PRpred = linreg.predict(x_poly)
PR = r2_score(Y, PRpred)

dectree = DecisionTreeRegressor()
dectree.fit(xtrain, ytrain)
DTpred = dectree.predict(xtest)
DT = r2_score(ytest, DTpred)

randreg = RandomForestRegressor(n_estimators=1000, random_state=10)
randreg.fit(xtrain, ytrain)
randpred = randreg.predict(xtest)
rfres = r2_score(ytest, randpred)

accuracy = [LR, PR, DT, rfres]
model = ['Logistic Regression', 'Polynomial Regression', 'Decision Tree', 'Random Forest']
```

Dataset Used:

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 factors 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 factor 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.



```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5043 entries, 0 to 5042
Data columns (total 28 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   color                      5024 non-null   object 
 1   director_name              4939 non-null   object 
 2   num_critic_for_reviews     4993 non-null   float64
 3   duration                   5028 non-null   float64
 4   director_facebook_likes    4939 non-null   float64
 5   actor_3_facebook_likes     5020 non-null   float64
 6   actor_2_name               5030 non-null   object 
 7   actor_1_facebook_likes     5036 non-null   float64
 8   gross                      4159 non-null   float64
 9   genres                     5043 non-null   object 
 10  actor_1_name               5036 non-null   object 
 11  movie_title                5043 non-null   object 
 12  num_voted_users            5043 non-null   int64  
 13  cast_total_facebook_likes  5043 non-null   int64  
 14  actor_3_name               5020 non-null   object 
 15  facenumber_in_poster       5030 non-null   float64
 16  plot_keywords              4890 non-null   object 
 17  movie_imdb_link            5043 non-null   object 
 18  num_user_for_reviews       5022 non-null   float64
 19  language                   5031 non-null   object 
 20  country                    5038 non-null   object 
 21  content_rating             4740 non-null   object 
 22  budget                     4551 non-null   float64
 23  title_year                 4935 non-null   float64
 24  actor_2_facebook_likes     5030 non-null   float64
 25  imdb_score                 5043 non-null   float64
 26  aspect_ratio               4714 non-null   float64
 27  movie_facebook_likes       5043 non-null   int64  
dtypes: float64(13), int64(3), object(12)
memory usage: 1.1+ MB
```

Data Preprocessing and Feature Engineering:

In our ambitious pursuit of precise IMDb score prediction, the critical phase of data preprocessing and feature engineering plays a pivotal role in honing our model's accuracy and interpretability. The initial dataset is a treasure trove of information, containing a plethora of attributes ranging from film duration, director and actor Facebook likes, earnings, budget, to engagement metrics like critic and user reviews. To optimize our IMDb score prediction endeavor, we meticulously curate a subset of these features that are most likely to exert an influence on IMDb scores.

One distinctive transformation applied is the conversion of IMDb scores into a categorical variable. This categorization allows us to classify films into distinct quality tiers: 'bad,' 'average,' 'good,' and 'excellent.' This categorization not only simplifies the prediction process but also facilitates a more intuitive understanding of audience reception.

To uncover the intricate relationship between film genre and IMDb scores, we extract the primary genre for each film and employ the Ordinal Encoder to convert it into a numerical feature. This transformation retains the hierarchical nature of different genres while enabling quantitative analysis.

Furthermore, we introduce a novel composite feature termed 'other_actors_likes,' which aggregates the Facebook likes of secondary actors, shedding light on the overall cast popularity, beyond the lead actor.

The dataset is rigorously cleaned to enhance data integrity, with the removal of outliers in the 'gross' variable and the meticulous handling of missing values. These steps contribute to the robustness of our IMDb score prediction model.

In summary, data preprocessing and feature engineering are foundational steps in our IMDb score prediction project. Through attribute selection, IMDb score categorization, genre encoding, and the creation of composite features, we prepare the dataset for more meaningful analysis and model development. These strategic adjustments enhance the clarity of our findings and bolster the overall precision of IMDb score predictions, providing valuable insights for decision-makers in the dynamic world of filmmaking.

Choice of Algorithm / Techniques:

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.

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.

Random Forest - Accuracy: 76.13%:

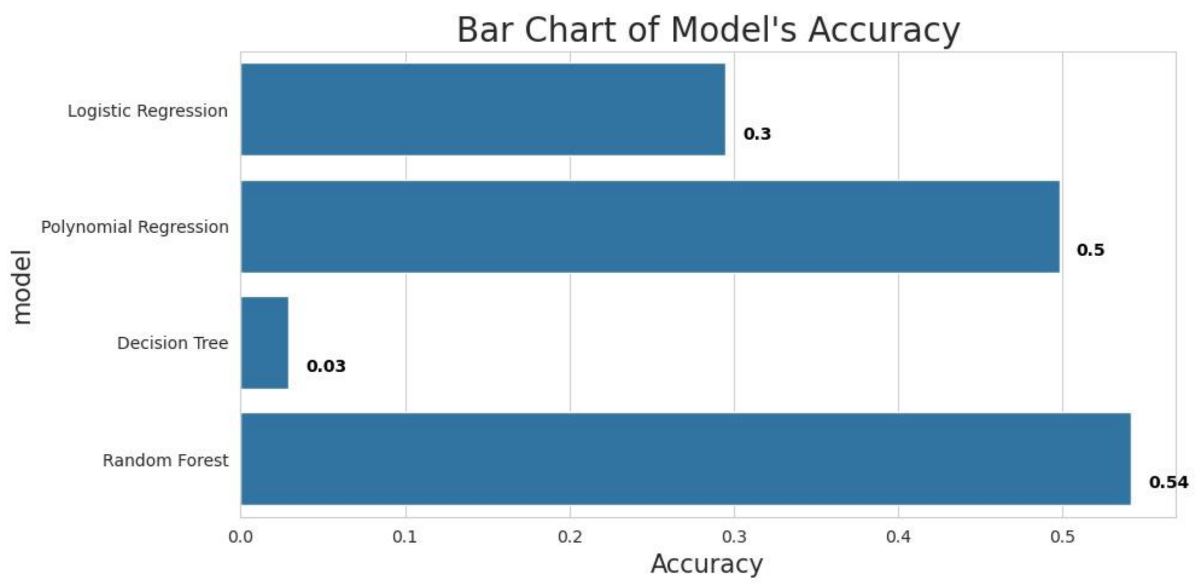
Data Preprocessing: Random Forest, being an ensemble method of decision trees, benefits from similar preprocessing steps as Decision Trees. Data cleanliness and handling outliers are important. Feature Extraction: Random Forest can handle a wide range of features effectively, but again, feature selection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

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.

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.

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.