**Predicting IMDb Scores**

**Innovation Phase - 2**

**Yenumula Hariteja**

**(Team Leader)**

**Innovation : Prediction of IMDb Scores**

**Introduction**

In this documentation, we explore the use of advanced regression techniques, specifically Gradient Boosting and Neural Networks, to improve the prediction accuracy for the project aimed at predicting IMDb scores. The dataset used for this project is sourced from Kaggle, containing features such as film title, genre, premiere date, runtime, IMDb scores, and available languages. The project's objective is to develop a machine learning model that accurately estimates the IMDb scores, facilitating user discovery of highly rated films matching their preferences.

**Problem Statement: A Brief**

The goal of the project is to create a machine learning model that can forecast the IMDb scores of movies on a platform like Films. It aims to do this by analysing various movie attributes such as genre, premiere date, runtime, and language. The primary objective is to build a model that can reliably predict a movie's popularity, making it easier for users to find highly-rated films that align with their tastes. To accomplish this, the project will involve tasks like preparing the data, creating new features from the existing ones, choosing an appropriate machine learning model, training it on the data, and assessing its performance.

**Dataset Details**

Here are some columns from the Netflix original films dataset that we use for our project to predict IMDb scores:

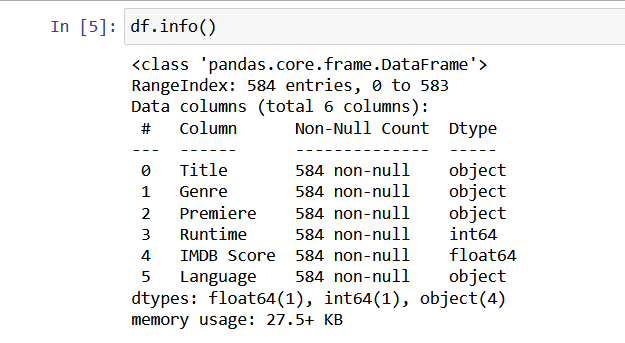
* Title: The title of the film.
* Genre: The genre of the film.
* Premiere date: The date on which the film was released on Netflix.
* Runtime: The length of the film in minutes.
* IMDB score: The film's rating on IMDb.
* Language: The primary language of the film.

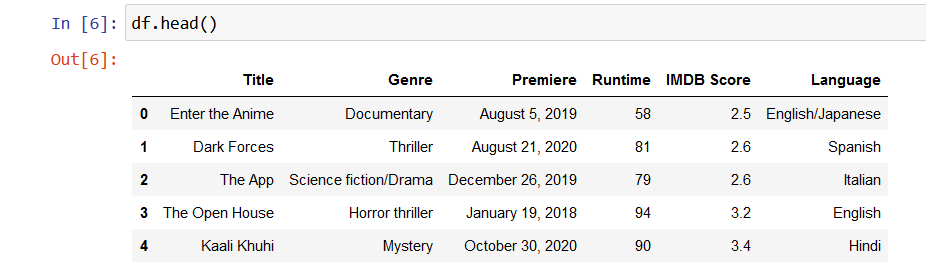
Here are the strategies we will be using for the different columns to predict IMDb scores:

**Genre**: Create a feature that represents the genre of the film. This could be a one-hot encoded feature, where each genre has its own column.

**Runtime**: Use the runtime of the film as a feature. This feature could be normalized to a scale of 0 to 1.

**IMDB score**: Use the IMDB score of the film as the target variable for your machine learning model.





**Libraries Needed**

The libraries we will be using in our project will be as follows:

1. **Pandas:** Pandas is a library for data analysis and manipulation. It can be used to read the Netflix original films dataset into a DataFrame, clean the data, and prepare it for machine learning.
2. **Scikit-learn:** Scikit-learn is a library for machine learning. It provides a variety of machine learning algorithms that you can use to train your model.
3. **Matplotlib:** Matplotlib is a library for data visualization. It can be used to create plots and charts to explore the dataset and visualize the results of your model.
4. **NumPy:** NumPy is a library for scientific computing. It can be used to perform mathematical operations on the data.
5. **Seaborn:** Seaborn is a library for statistical data visualization. It provides a high-level interface to Matplotlib that makes it easier to create informative and attractive plots.
6. **XGBoost**: XGBoost is a library for gradient boosted decision trees. It is a very powerful machine learning algorithm that is often used for regression tasks, such as predicting IMDb scores.

**Training And Testing**

To train and test the data, we will follow these steps:

* Prepare the data
* Choose a machine learning algorithm
* Train the model
* Evaluate the model
* Tune the model
* Deploy the model

**Prepare the data**

The first step is to prepare the data. This involves cleaning the data, removing outliers, and splitting the data into training and testing sets. To clean the data, you need to remove any errors or inconsistencies in the data.

To remove outliers, we will identify any data points that are significantly different from the rest of the data. Outliers can skew the results of your machine learning model, so it is important to remove them.

To split the data into training and testing sets, we will randomly divide the data into two sets. The training set will be used to train the model, and the testing set will be used to evaluate the model.

**Choose a machine learning algorithm**

We use algorithms such as linear regression, support vector machines, and decision trees.

**Train the model**

Once the algorithm is chosen, we need to train the model on the training set. This involves feeding the model the training data and allowing it to learn the patterns in the data.

The training process can take some time, depending on the size of the dataset and the complexity of the algorithm.

**Evaluate the model**

Once the model is trained, we need to evaluate its performance on the testing set. This involves feeding the model the testing data and measuring how well it can predict the target variable.

The evaluation metrics that you use will depend on the specific task that you are trying to solve. For example, if you are trying to build a model to predict IMDb scores, we want to use the mean squared error (MSE) or the root mean squared error (RMSE).

**Tune the model**

If the model is not performing well enough, we will try to tune the hyperparameters of the model. Hyperparameters are the parameters that control how the algorithm works. Tuning the hyperparameters can improve the performance of the model. To tune we use more sophisticated method, such as Bayesian optimization.

**Deploy the model**

Once the model is trained and tuned, we deploy it to production. This means making the model available to users so that they can use it to make predictions.

There are many different ways to deploy a machine learning model. We can deploy the model to a web service, a mobile app, or an embedded device.

**Traditional Regression Approaches**

Before delving into advanced techniques, it's essential to understand the traditional regression approaches used in this project. These include Linear Regression, Lasso Regression, and Ridge Regression. These models form the baseline for comparison.

**Linear Regression**

Linear regression models the relationship between the IMDb score and the features as a linear equation. This approach assumes a linear relationship between the features and the target variable.

**Lasso and Ridge Regression**

Lasso and Ridge regression are used to handle multicollinearity and overfitting. Lasso adds an L1 regularization term, while Ridge adds an L2 regularization term to the linear regression.

**Advanced Regression Techniques**

Gradient Boosting

Gradient Boosting is an ensemble learning technique that builds multiple decision trees sequentially. Each tree corrects the errors of the previous one. The use of Gradient Boosting provides a powerful tool to model complex relationships in the dataset.

**Innovation: Feature Importance Analysis**

One innovative approach with Gradient Boosting is to perform feature importance analysis. By understanding which features have the most significant impact on IMDb scores, we can focus on optimizing those features. Feature importance can be visualized and used to prioritize data collection or engineering efforts.

**Hyperparameter Tuning**

Utilizing techniques like Grid Search or Random Search to optimize hyperparameters can greatly improve the performance of the Gradient Boosting model. These hyperparameters may include the learning rate, number of estimators, maximum depth of trees, etc.

**Neural Networks**

Neural Networks, particularly deep learning techniques, can capture highly complex and non-linear relationships within the data. They consist of multiple interconnected layers of neurons, each performing its own transformation of the input data.

**Innovation: Embedding Layers**

In the case of categorical features such as film genre and language, we can utilize embedding layers. These layers learn meaningful representations of these categorical variables and can improve the model's ability to capture subtle nuances in these features.

**Model Stacking**

Neural networks can be combined with traditional regression models to form a stacked model. This approach leverages the strengths of both models, potentially improving overall prediction accuracy.

**Implementation Steps**

1. **Data Preprocessing**:

Cleaning and transforming the dataset, handling missing values, and encoding categorical variables.

2. **Feature Engineering:**

Create new features, derive meaningful insights, and perform feature scaling and normalization.

3. **Model Selection:**

Choose between Linear Regression, Lasso, Ridge, Gradient Boosting, Neural Networks, or a combination of these models based on cross-validation results.

4. **Training and Evaluation:**

Train the selected models and evaluate their performance using metrics like Mean Squared Error (MSE), R-squared, and cross-validation techniques.

5**. Innovation Integration:**

Implement the proposed innovations, such as feature importance analysis for Gradient Boosting and embedding layers for Neural Networks.

6. **Hyperparameter Tuning:** Optimize the hyperparameters of Gradient Boosting and Neural Networks to maximize prediction accuracy.

7. **Model Ensemble:**

Consider ensemble techniques to combine the strengths of various models, if necessary.

8. **Monitoring and Maintenance:**

Regularly monitor the model's performance and update it with new data as IMDb scores change.

**Conclusion**

By incorporating advanced regression techniques like Gradient Boosting and Neural Networks, this project aims to significantly improve the prediction accuracy for IMDb scores of Netflix original films. The innovative approaches presented here, such as feature importance analysis and embedding layers, have the potential to enhance the model's performance and help users discover highly rated films that match their preferences effectively. It's essential to document and iterate on these innovations, keeping the model up-to-date with changing IMDb scores and user preferences.