IMDb Movie Rating Prediction Using KNN & Random Forest

1 Introduction

The film industry is a major global business, generating excitement worldwide about a movie's box office success and popularity. Leveraging this fascination, this project utilizes a rich dataset from IMDb, the world's leading online resource for movie, TV, and celebrity information, which includes comprehensive data on over 5000 movies.

2 Data

IMDb 5000 Movie dataset[1] offers extensive details such as movie titles, duration, director and actor names, social media engagement, keywords, genres, countries of production, gross revenues, and IMDb scores. In total, there are 27 features given, including 10 categorical features and 17 numerical features (See Table 1).

Table 1: Features Description

Features	Type
id	numeric
director_name	category
num_critic_for_reviews	numeric
duration	numeric
director_facebook_likes	numeic
actor_3_facebook_likes	numeric
actor_2_name	category
actor_1_facebook_likes	numeric
gross	numeric
genres	category
actor_1_name	category
movie_title	category
num_voted_users	numeric
cast_total_facebook_likes	numeric
actor_3_name	category
facenumber_in_poster	numeric
plot_keywords	category
num_user_for_reviews	numeric
language	category
country	category
content_rating	category
title_year	numeric
actor_2_facebook likes	numeric
movie_facebook_likes	numeric
title_embedding	numeric
average_degree_centrality	numeric
imdb_score_binned	numeric

[1]https://www.kaggle.com/datasets/carolzhangdc/imdb-5000-movie-dataset

3 Method

This project aims to avoid increase of dimensionality, along with maximizing usage of categorical features in the dataset, by cleaning and transforming raw data and selection of important features.

3.1 Data Preprocessing

3.1.1 Handling Categorical Features

The main method is utilizing visualization on the distribution of instances on each feature and choosing top 4 features out, while dropping the other (See ii, iii).

i. Genres

Genres are split into unique keywords, resulting in a total of 22 distinct genres. Each genre is encoded using a binary representation, similar to one-hot encoding, aims to transform categorical data into numerical. This approach is chosen to address the aims of reducing dimensionality, preventing the exponential increase in dimensionality that could occur with other encoding methods (binary, countvec, dotvec), where the vector has to be flatten which increases the overall dimensionality.

ii. Content Rating

Figure 1 highlights the top most common content ratings in the dataset. The top 4 namely `R, PG, PG-13, G` are chosen. These ratings are retained for their substantial representation in the dataset, while less frequent ratings are excluded to streamline the feature set.

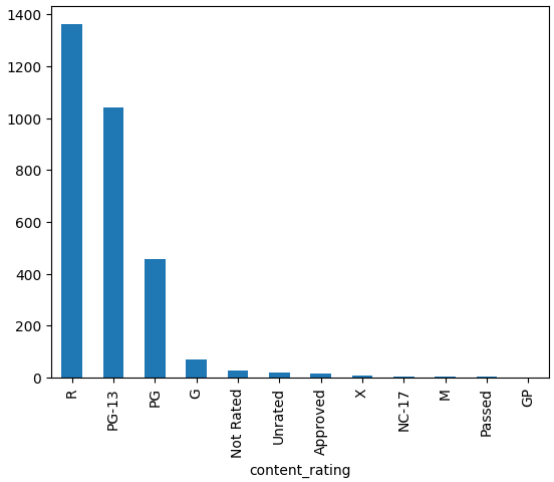

Figure 1: Content rating

iii. Country

Figure 2 illustrates the top four most common countries represented in the dataset, including `USA, UK, France, Germany`. Similar to Content Rating, retain these countries due to their significant presence.

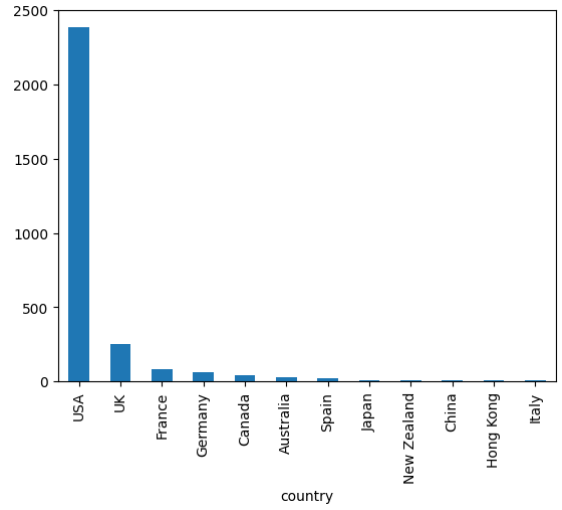

Figure 2: Country

iv. Language

Demonstrated in Figure 3, English predominates approximately 95% of the entries. Therefore, remove this feature from the dataset to avoid redundancy from non-informative features.

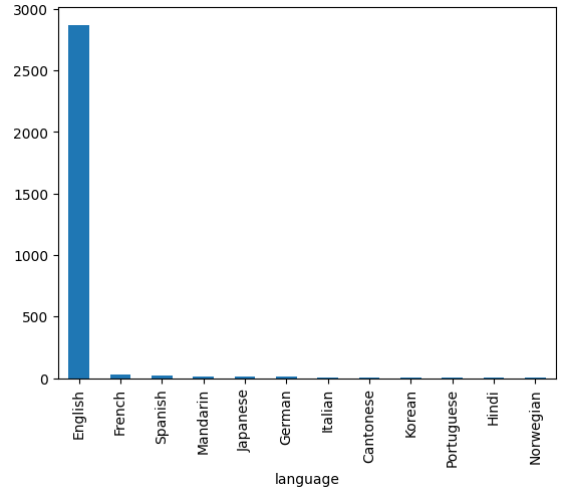

Figure 3: Language

v. Other categorical features

The remaining categorical columns include:

`director_name, actor_1_name, actor_2_name, actor_3_name, movie_title, plot_keywords`

Encoding these features would significantly increase dimensionality due to their uniqueness—each name and title is distinct. Moreover, because these features have unique values, there is a high likelihood that many will not appear in the test set. Consequently, this project excludes these columns to maintain a manageable feature set and enhance the model's robustness and efficiency.

`title_embedding` (i.e., the fasttext vector of `movie_title`) is also removed as `movie_title` is excluded.

3.1.2 Feature Selection 1: Correlation

Figure 4 illustrates that the features:

`face_number_in_posters,`
`actor_1_facebook_likes,`
`actor_2_facebook_likes,`
`actor_3_facebook_likes,`
`cast_total_facebook_likes`

exhibit low correlations with the target variable, taking absolute values of the correlation results in 0.06, 0.08, 0.08, and 0.06, respectively. Given these weak correlations, we apply a threshold of 0.1 to determine feature relevance. Features falling below this threshold are considered to have minimal impact on the model's predictive capability, hence chosen to remove.

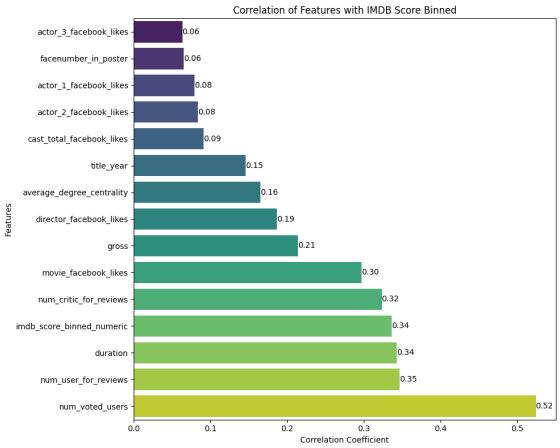

Figure 4: Correlation: Features vs IMDb Score

3.2 Data Scaling

In this project, data scaling is crucial due to the highly skewed nature of the dataset (Figure 5). Standardization is preferred because it is robust to outliers, maintaining their influence in the dataset. This retention of outlier weight is particularly beneficial for the LASSO method used in feature selection (See section 3.3). Additionally, standardization aids in handling the skewed data by centering the distribution of features, thereby mitigating the effects of skewness and enhancing the performance of models.

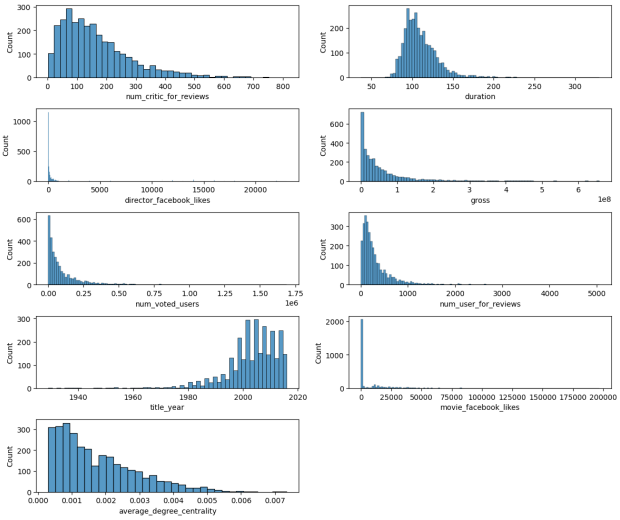

Figure 5: Numerical Features Distribution

3.3 Feature Selection 2: LASSO

Continuing from section 3.1.2 on feature selection, LASSO is chosen for its efficiency in managing high-dimensional data, while making optimal use of categorical variables. LASSO's ability to enhance model performance by selectively penalizing less significant features. Additionally, the standardization phase ensures that each feature contributes equally to LASSO's regularization process, preventing features with larger scales from dominating, and robust against outliers, synergizes with LASSO's regularization technique, maintaining a balance where outliers do not unduly influence the model. After LASSO resulting in total 25 selected features:

`USA, Germany, R, PG-13, Biography, Animation, History, Action, Documentary, Sci-Fi, Horror, Drama, Comedy, Thriller, Fantasy, Family, War, Adventure, num_critic_for_reviews, duration, gross, num_voted_users, num_user_for_reviews, title_year, movie_facebook_likes`

4 Model Implementation & Hyperparameter Tuning

The train dataset is split into training and validation sets using stratified K-fold cross-validation with n = 15 splits. The rationale for employing stratified K-fold in this scenario stems from the distribution of the IMDb scores observed in Table 2, dominance of label **2** (>50%). This approach prevent model overfitting by ensuring that each training and validation fold is a good representative of the whole dataset.

imdb_score_binned	% in dataset
0	1%
1	8%
2	61%
3	26%
4	4%

Table 2: Distribution of labels

4.1 K Nearest Neighbour

The K-Nearest Neighbors (KNN) algorithm was chosen to classify IMDb scores due to its efficiency in handling classification problems where the relationship between feature patterns and class labels is not linear.

Initial model Initially, our KNN classifier was set with n = 5 neighbors, distance calculated using Euclidean distance, which results in accuracy of 67.07%.

Hyperparameter Tuning: To optimize our KNN model, we employed a systematic approach using GridSearchCV with StratifiedKFold cross-validation (15 splits). The parameters grid was defined with variations in the number of neighbors (1 to 20), weight types ('uniform' vs. 'distance'), and the distance metric (Manhattan and Euclidean distances).

Result after tuning: The results from the grid search revealed the optimal parameters as {*'n_neighbors': 18, 'p': 1, 'weights': 'distance'*}, indicating increase in number of

neighbour, distance-weighted voting, and the Manhattan distance metric were most effective for training KNN model. Manhattan distance tends to perform better than Euclidean distance in high-dimensional settings because it sums the absolute differences across dimensions, making it less sensitive to large discrepancies in any single dimension (Gohrani, 2019). With these optimal parameters, the model's accuracy improved to 70.27%, an increase of 3.2% from the initial model.

4.2 Random Forest

The core idea behind Random Forest is to build a large number of individual decision trees that operate as an ensemble. The ensemble nature of Random Forest helps in handling the limitations observed with KNN, particularly in terms of efficiency and handling high-dimensional data. Each tree in the forest considers a random subset of features when making decisions, which significantly reduces the risk of overfitting.

Initial model: Initially, the Random Forest classifier recorded an accuracy of approximately 74.30%.

Hyperparameter Tuning: To refine this performance, an extensive hyperparameter tuning was implemented using scikit-learn's RandomizedSearchCV. This approach was chosen over scikit-learn's GridSearchCV due to its efficiency in navigating large hyperparameter spaces more quickly. The tuning process, facilitated by StratifiedKFold cross-validation, aimed to optimally adjust several parameters, including the number of trees (*n_estimators*), the maximum number of features to consider at each split (*max_features*), the maximum depth of each tree (*max_depth*), and the minimum number of samples required both to split a node (*min_samples_split*) and at each leaf node (*min_samples_leaf*).

Result after tuning: After tuning, the model demonstrated a slight improvement in accuracy, reaching up to 74.60%. While this increase may appear modest (0.3%), it underscores the model's enhanced robustness and precision in handling the classification of IMDb movie scores (See section 5.2).

5 Model Evaluation and Analysis

5.1 K Nearest Neighbour

5.1.1 Evaluation

The average accuracy of 70.27% initially suggests a reasonable efficacy in general predictions. The bias of 0.5912 (see Table 3) suggests a consistent underestimation or overestimation across predictions. This could be due to the KNN model's sensitivity to the choice of 'k', which might not be optimally tuned, or the model's inherent limitations in handling overlapping feature spaces where different classes share similar attributes.

In algorithms like KNN that rely heavily on distance metrics, the scale and type of features significantly impact performance. If features contributing to the KNN model fail to encapsulate critical distinguishing characteristics (e.g., genre-specific attributes, temporal aspects like release year trends), the model's ability to discern between close but distinct classes will be impaired.

The reported precision and recall (See Table 3), are quite balanced, indicating that the model is relatively consistent in identifying true positives but could be misclassifying a significant portion of positives, especially in minority classes. The balance also suggests that the model does not heavily favor avoiding false positives over identifying all positives, or vice versa. Furthermore, the F1 score illustrates that the model might not be effectively balancing the precision-recall trade-off, possibly due to it being influenced more by the majority class or by classes that are easier to predict.

Metric	Result
Accuracy	70.27%
Precision	67.91%
Recall	70.27%
F1	65.53%
MSE	0.5912

Table 3: KNN Evaluation Metrics

5.1.2 Error Analysis

The confusion matrix (Figure 6) further elucidates the model's performance nuances.

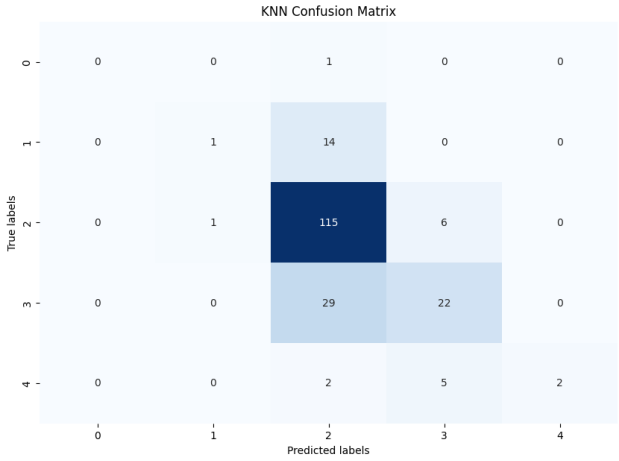

Figure 6: Random Forest Confusion Matrix

It reveals a conspicuous absence of predictions for the lowest score category (class 0), and significant confusion between adjacent score categories, particularly between classes 1 vs 2 (14 out of 15 instances classify incorrectly from 1 to 2) and 2 vs 3 (29 out of 51 instances is classify as 2 instead of 3). This pattern may indicate the model's limitations in distinguishing between subtly differing scores, potentially due to the feature set's inability to capture finer distinctions in movie quality. The most accurate predictions are observed in class 2, with true positive of 94% of the time.

5.2 Random Forest

5.2.1 Evaluation

The Random Forest model demonstrates an average accuracy of 74.60%, which is not significantly higher than KNN, but shows a fairly better performance in handling of the complexities within the dataset (see metrics on Table 4). The precision is reported at 73.03%, indicating a moderate amount of false positive predictions. The recall rate matches the accuracy at 74.60%, indicating the model's strength in identifying positive instances. Precision and Recall are closely aligned, indicating a good balance between the number of correct positive predictions made and the model's ability to capture the majority of actual positive cases. The model exhibits a bias of 0.5316, indicating that there might still be some level of error in the model's predictions that is systematic across different tests. This means that while the categories are ranked, the differences between them are not quantitatively even. Standard Random Forest models treat these as categorical without considering the inherent order, potentially leading to misinterpretations or oversimplified distinctions between adjacent scores. Misclassification between such scores may contribute to the observed bias, as the model does not effectively quantify the nuanced differences between close ordinal categories.

Metric	Result
Accuracy	74.60%
Precision	73.03%
Recall	74.60%
F1	71.17%
MSE	0.5316

Table 4: Random Forest Evaluation Metrics

5.2.2 Error Analysis

Further on error analysis, the confusion matrix (Figure 7) provides a detailed look at the model's predictive accuracy across specific classes.

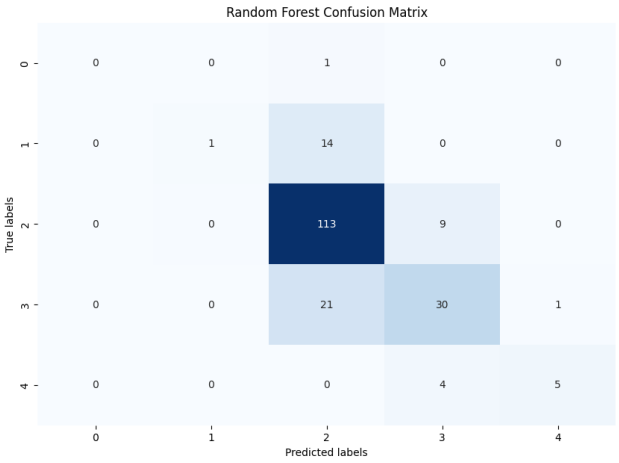

Figure 7: Random Forest Confusion Matrix

Similarly to KNN, class 0 has 1 false positive , this can be explained from the dataset, where the number instances trained being 0 is significantly low, hence low information trained on class 0. However, it shows that Class 2 is well-predicted with 113 true positives but exhibits some confusion with Class 3, where 9 instances of Class 2 are misclassified. Class 3 shows a substantial confusion, particularly with Class 2, misclassifying 21 instances as belonging to Class 2. Class 4, shows confusion with Class

3, as it classifies 4 out of 9 instances belonging to class 3, indicating significant overlap in features or inadequate distinctiveness between these classes. Furthermore, the instance distribution across the dataset dominantly trains on label 2 (See table 2), hence the most accurate predictions are observed in class 2, whereas misclassify on others.

5.3 Analysis on Test Data

Random Forest was trained and used to apply on the test data. A focused evaluation on 50% of the test set reveals that the accuracy slightly decreases to 72%, compared to the average accuracy of 74.67% observed across the entire dataset, suggesting no or minor data leakage during training, as well as the model trained is well-fitted.

6 Conclusions

Overall, the Random Forest model outperformed the KNN classifier in terms of accuracy and handling dataset complexities, demonstrating its robustness and suitability for IMDb movie classification tasks. Future work could involve exploring advanced feature engineering, balancing techniques for underrepresented classes, and integrating more sophisticated ensemble (ordinal and classification) methods to further enhance model performance.

References

Kunal Gohrani. 2019. Different types of distance metrics used in machine learning.