# IMDB Movie Rating Prediction

COMP30027 Assignment 2

## 1. Introduction

Can we accurately predict IMDB movie ratings from multimodal predictor variables? To answer this question, we recall the largely theoretical lecture concepts surrounding data representation, classifier construction, evaluation and error analysis. By applying preprocessing techniques to categorical data and evaluation of numerical data, we aim to predict movie ratings.

To predict movie ratings using multimodal data, we conduct an investigation into which features have the strongest predictive power, and evaluate their effectiveness in accurately predicting movie ratings. Further enhancements in the form of feature engineering, feature selection, model selection, hyperparameter tuning and ensemble methods also aid in our predictive capabilities. Specifically, we explore the use of K-NN, random-tree forests, SVM, and Logistic regression as input algorithms.

Our aim is to explore the dataset, visualise results, and critically analyse the performance of the classifiers and achieve high accuracy. Furthermore, we also hope to understand the underlying reasons for the performance of different methods and how they relate to the theoretical concepts discussed in our machine learning course.

## 2. Feature Analysis

### 2.1 Dataset

We used data, "IMDB 5000 Movie Dataset" [1] from Kaggle. The selected features were provided by staff and can be found in the "project_data" directory as columns for "train_dataset.csv".

We split these features into 3 categories: numerical (num_critic_for_reviews), categorical (language, country) and name features. These can be found in the "README.MD" on my GitHub repo [2] with descriptions of each feature.

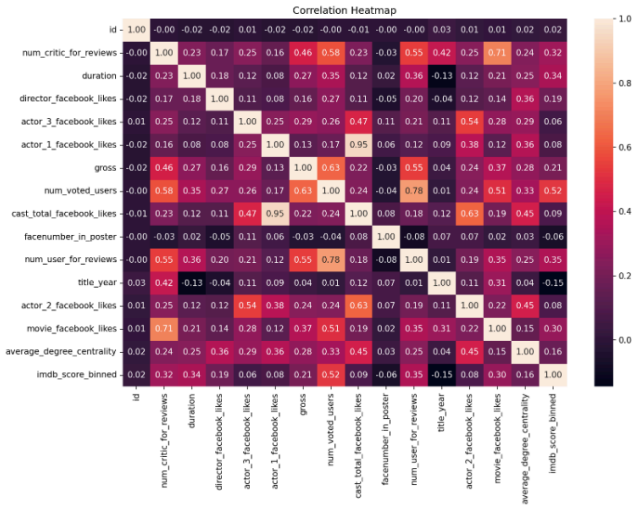


**Figure 1 -** Correlation heat map for numerical values in training dataset

### 2.2 Feature Engineering

Inspecting the features closely provided valuable insights into how I could clean the data prior to learning. Director names proved to be high-dimensional with many unique cases. To save computational power whilst preserving the variance, I collated directors with only one movie to a single value 'Other'. For 'plot_keywords' and 'title_embeddings', I opted to make use of the doc2vec and fast-text embeddings respectively with epochs = 100. Genres for each entry were separated by '|', and so I converted these into dummies and used One-Hot-Encoding to append these features to the dataset. Considering the results from the frequency graph (Figure 2), content rating was mapped into broader categories of 'PG', 'R', 'M', 'Unrated' and 'Other'. The top 2 occurring countries

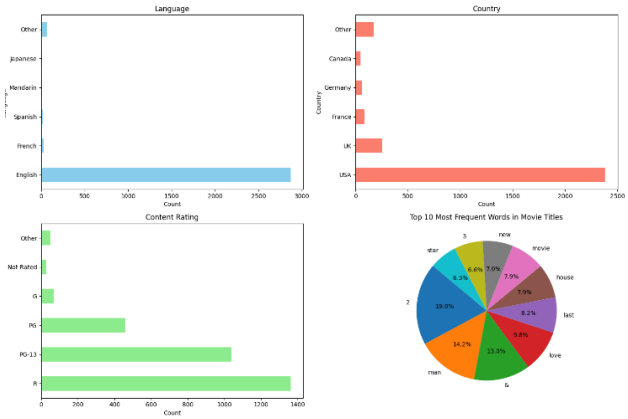


**Figure 2 -** Frequency of categorical data

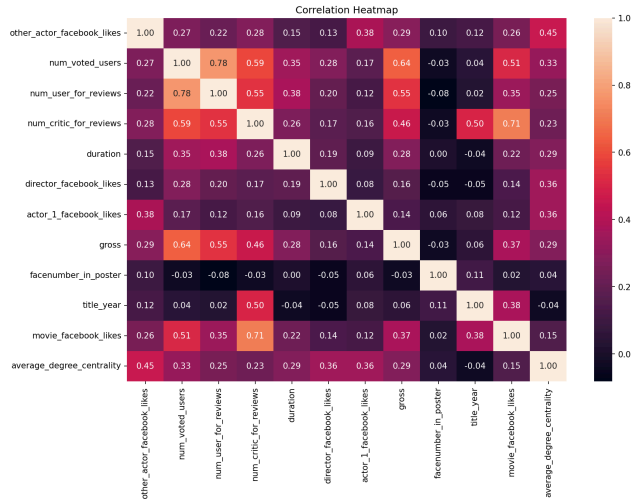


**Figure 3 -** Correlation heat map for numerical values after feature engineering

were the USA and UK, and all other countries were grouped as 'Other'. For movie titles, I used NLTK for tokenization and removed stop words.

Analysing the data showed that there was one missing value in the language feature and it's usually good practice to handle missing values by imputing using most-frequency or mean. However, the frequency graphs (Figure 2) show that 96% of movies are 'English' and so I opted to remove the language feature entirely. After removing movies released before 1980, I had 2921 entries remaining in my training dataset. Then, by observing the correlation heatmap (Figure 1), where actor_1_likes and cast_total_likes = 0.95, I dropped 'id' and 'cast_total_likes'. Finally, I summed the facebook likes of actor 2 and 3 into one feature. In the end, it arrived at 42 features from the original 26 and all correlation scores were less than 0.8 (Figure 3).

### 2.3 Data Preprocessing

Many machine learning models only accept numerical data and even after feature engineering, there was still categorical data remaining (movie_title, actor_3_name, etc.). Therefore, the data preprocessor serves to standardise all numerical features, condense sparse matrix embeddings by calculating mean values, encode categorical data using OneHotEncoding and vectorise features like movie title and names using the Tf Idf metric. After fitting my preprocessor to the data, the transformed dataset had dimensions of 5978, predominantly due to the high number of unique entries for movie titles.

## 3. Methods

In this section, I will draw upon my findings from feature analysis and discuss the models I chose to implement, and how they were used in tandem to find the best performing model.

### 3.1 K-nn model

For a baseline model, I used a k-nn algorithm, which is simple to implement and doesn't involve any training steps as it simply tags new data based on what it has been provided with. It is one of the most common, fundamental and simple methods used for both classification and regression tasks.

It is however prone to the curse of dimensionality, and also doesn't perform well on imbalanced data as it tends to favour the majority class. The choice of k (number of neighbours) can improve performance but in general, underperforms compared to other models that can learn more complex equations of the data.

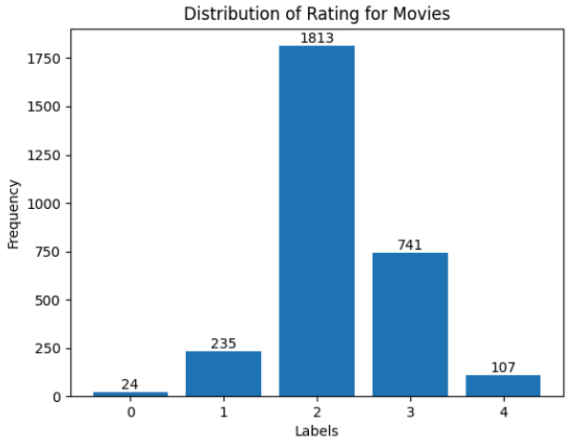


**Figure 4 -** Distribution of rating for movies in training dataset

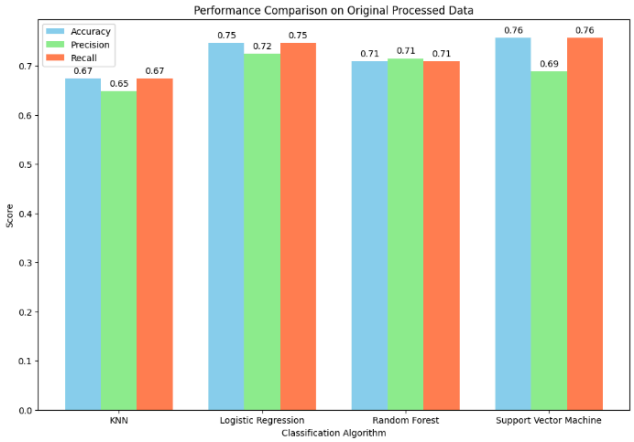


**Figure 5 -** Performance comparison on original processed data models

### 3.2 Logistic Regression

Logistic regression is a classification model that forms a linear decision boundary and learns the coefficients for each independent variable in predicting the target class. Specifically, I used the L-BFGS solver with max iterations to convergence set to 1000. To mitigate overfitting that might occur from fitting a linear decision boundary on the data, I also used regularisation methods to add an l2-norm penalty to the cost function to penalise larger coefficients. In general, this improves the predictive abilities of the model. Finally, the coefficients learned by the model can provide valuable insights into the importance and influence of each feature on the movie rating.

### 3.3 Random Tree Forest

Since the data could fit well to a non-linear model, I also decided to experiment with random forest, an ensemble method that aggregates the results of multiple decision trees. It uses bagging to split features into a random subset whilst selecting the best feature for each node of a decision tree. It is generally a very strong performer for non-linear problems and is robust to overfitting. However, it can be sensitive to many parameters for example (number of subtrees, min-splits, min-leaves, etc.), and so extensive testing and tuning is required to use this model optimally.

### 3.4 Support Vector Machine

Support Vector Machine (SVM) is a widely used machine learning model and is well-suited for classification tasks, and is effective in high-dimensional spaces. SVM finds the optimal hyperplane that separates classes in the feature space, whilst maximising the margin between classes. For this specific implementation, I used the radial basis function (rbf) to capture more complex relationships.

## 4. Investigations and Results

### 4.1 Metrics

In this investigation, I used accuracy and classification reports as my main metrics. These reports were used during model selection and accuracy for a detailed look into the effects of hyperparameter tuning.

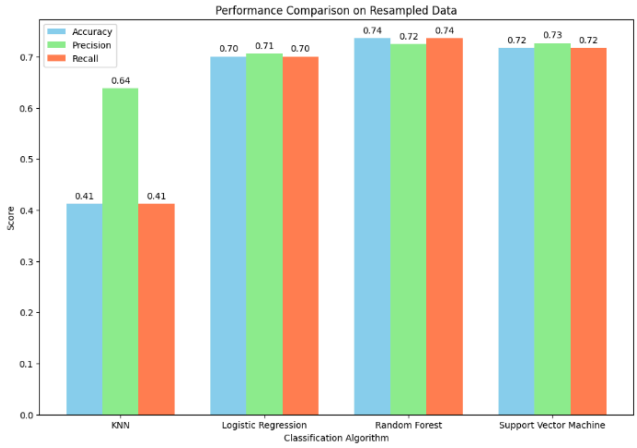


**Figure 6 -** Performance comparison on SMOTE resampled data

## 4.2 Base Models

First, I defined classifiers for each input algorithm and obtained classification reports for each of these models. When we trained all features on these (default parameters from sci-kit learn.), SVM yielded the highest accuracy of 0.7568. However, it's also worthy to take note of its relatively lower weighted precision score of 0.69. Hence, it was necessary to investigate logistic regression as well due to its similar accuracy score of 0.7465. Results shown in Figure 5.

Before further investigation, I also resampled data using SMOTE [4] to see if oversampling could provide benefits to each model due to the imbalanced nature of the dataset seen in Figure 4. From the results of the resample data (Figure 6), we see that the performance of logistic regression, random forest and SVM remain relatively similar. However, the accuracy and recall of K-NN dropped drastically by 0.26 each. This could be due to the synthetic data points introduced using SMOTE and thus increasing the degree of noise. Thus, this technique was not implemented for further analysis.

| Model | Hyperparameters | Accuracy |
|---|---|---|
| Logistic Regression | C = 0.5<br>Num. features = all | 0.7434 |
| SVM | C = 4 | 0.7424 |
| Random Forest | Max Depth = max<br>Max Features = sqrt | 0.7064 |
| Stacking | N/A | 0.7552 |

**Table 1 -** Accuracy of models after hyperparameter tuning

## 4.3 Tuning Models

I set the results from k-nn as a baseline, and used repeated stratified K-fold cross-validation with 5 splits and 3 repeats to tune hyperparameters. I tested for different values during feature selection, dimensionality reduction, class weighting and different regularisation techniques and values. For logistic regression, the optimal c was 0.5 with regular class weighting. For SVM, we set the kernel to rbf and the optimal regularisation parameter was 4. For random tree forest, we used 100 trees and set the max depth to max and max features to sqrt of total number of features. The results are shown in Table 1.

I found that all features after preprocessing contribute to the prediction of movie ratings. The current results show that logistic regression has the best performance, accurately predicting 74% of the IMDB movie ratings.

## 4.4 Ensemble Methods

Using the predictions of each of the three base classifiers as an input to a meta-classifier is a well-known method known as stacking. I decided to use logistic regression as the meta classifier in hopes of making use of its ability to combine heterogeneous classifiers with varying performance.

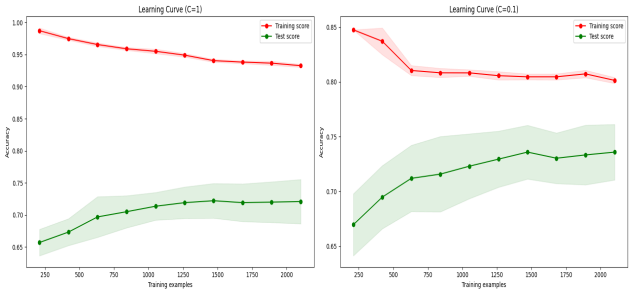


**Figure 7 -** Learning curve using logistic regression with c = 0.1 and c = 1

This classifier was able to achieve an accuracy of 0.7552 on the test set, which is a slight improvement over the base classifiers.

# 5. Discussion and Critical Analysis

## 5.1 Best Model

Adding regularisation improved overall model performances, and feature selection after processing the data didn't seem to benefit performance. Logistic regression and stacking have similar accuracy, but stacking is significantly slower and ultimately depends on individual classifier outputs. Considering accuracy, efficiency and constraints of resources, I chose logistic regression as my best model.

## 5.2 Learning Curve

The learning curve analysis offers valuable insight into model performance in relation to size of training data. From Figure 7, we observe that at c = 1, with the optimal number of training samples (1500) there is a margin of approximately 0.21 between accuracy scores of the training set and test set. This suggests that our model could be overfitting the data, and thus the performance on unseen data is much worse. However, when using a lower value of c and the same split, we prioritise smoother margins and now have a margin of 0.12, suggesting that our generalisation capabilities have improved.

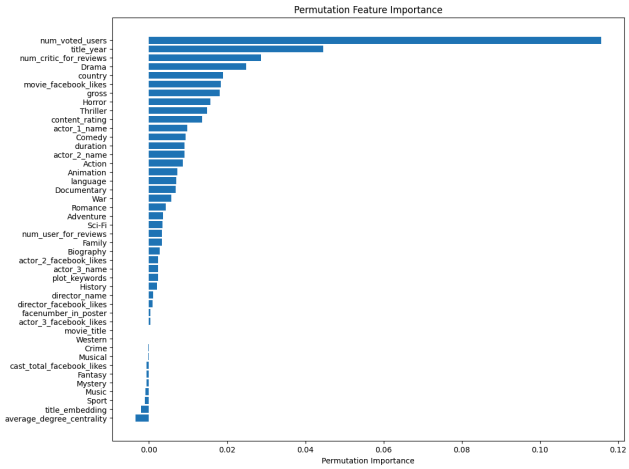


**Figure 8 -** FI using tuned logistic regression model

## 5.3 Permutation Feature Importance

To evaluate the contributions made by different features, I used permutation feature importance (FI), which measures the contributions of each feature to a model's statistical performance. It randomly shuffles the values of a single feature and observes the degradation to the model's performance. For features j {1…, p}, FI is defined as:

$$FI_j = e_{perm} - e_{orig}$$

Where err-orig is the model error with all original features, and err-perm is the model error when feature j is permuted.

For logistic regression FI (Figure 8), genres such as drama, horror, thriller, comedy and action have high FI. This could be due to the main-stream factor of these genres. By contrast, niche genres such as sport, music and musical are lower. Number of voted users is the most important factor that can be used to predict movie ratings by quite a margin with a FI of 0.12. In addition to this, title year, number of critic reviews and country are also important. FI simply determines the magnitude of change, not necessarily the direction of the feature's effect on movie rating. Thus, I fit a new subset of features (top 15 FI) on each base classifier and observed the differences to

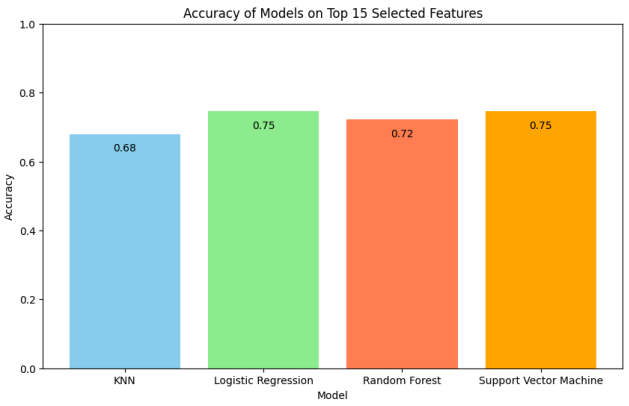


**Figure 9 -** Accuracy of base classifiers using subset of features (top 15 FI)

their original performance (Figure 9). Similar to the original features, logistic regression and SVM performed the best but did have any noticeable improvements as they remained at an accuracy score of around 75%.

## 6. Conclusion

By feature engineering the dataset and processing multimodal data from the IMDB movie dataset into a logistic regression model, we could predict movie ratings with an accuracy of 75%, which is better than most models for predicting human behaviour [5]. Among all the features, statistics for the number of votes and reviews had the highest predictive power on predicting movie ratings. Certain genres like drama or horror were able to explain ratings more so than others like sports and musical. Further analysis of models using a subset of features with FI could not significantly improve performance, but feature engineering did. Thus, to achieve better results in future studies, it would be wise to add new features, such as movie budget, which could help us evaluate commercial success of movies in relation to gross earnings. Another area of future research could be more extensive use of ensemble methods. Although stacking was able to achieve similar results to our base classifiers, additional tuning and different preprocessing techniques could boost accuracy and better combine information from all three base classifiers.

Word count: 1946

## 7. Reference

[1]https://www.kaggle.com/datasets/carolzhangdc/imdb-5000-movie-dataset
[2]https://github.com/JZYNX/ML-assignment-2
[3]https://towardsdatascience.com/finally-why-we-use-an-80-20-split-for-training-and-test-data-plus-an-alternative-method-oh-yes-edc77e96295d
[4]https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html
[5]https://pure.manchester.ac.uk/ws/portalfiles/portal/256978591/behaviour_prediction_review_manuscript.pdf