COMP30027 Report

Anonymous

1. Introduction

The objective of this report is to thoroughly examine four supervised learning methods for predicting IMDB movie ratings based on different predictor variables. These predictors include movie-title, duration, director and actor(s) names and Facebook likes, keywords, genre, country, budget, and others.

The report introduces the four machine learning models that were built and trained on the dataset along with the preprocessing steps required to clean the data. Then the report delves into statistical analysis comparing the performance of different models, and analyzing how each model performs with different hyperparameters and holdout strategies.

Lastly the report emphasizes the errors of different models and discusses their behaviour when encountering unseen data.

2. Methodology

Four models: ZeroR, Decision Tree, Logistic regression, and K-NN were trained and built for this project. The dataset consisted of 27 features including both numeric and non-numeric ones. The dataset needed preprocessing since the chosen models can only work with numeric features.

Firstly, four features: 'director_name', 'actor_1_name', 'actor_2_name', and 'actor_3_name' were dropped based on intuition. The underlying explanation behind this decision was that the dataset already included Facebook likes for these entities, which likely have more significant impact on popularity than just their names. This is presumed to have a greater effect on the target variable 'imdb_score_binned'.

Secondly, various embedding techniques were applied to specific non-numeric features. For example, 'genres' and 'plot_keywords' utilized doc2vec whereas 'movie_title', 'language', and 'country' employed FastText embedding. While there are several approaches to utilizing these embeddings, it was decided to take their mean.

The subsections below will discuss how each model was further engineered, including any holdout approaches or hyperparameter tuning.

2.1 Logistic Regression

Further feature engineering was done by removing the features with weights close to zero. Subsequently, a grid search was conducted to determine the optimal maximum number of iterations. The training dataset was then into train and test sets. Various optimization algorithms were employed. Due to the target variable consisting multiple classes, two types of multiclass logistic regression were used: one-vs-one and one-vs-rest. The logistic regression models were trained on the training split and evaluated on the testing split to determine the maximum accuracy and identify the best performing optimization algorithm. Additionally, 10-fold-cross-validation was used to ensure the reliability of results

2.2 K-NN

Like logistic regression, redundant features were removed to reduce the cure of high dimensionality. Likewise, the training dataset was partitioned into separate training and testing sets. The optimal value of k for k-NN was determined by identifying the point where the accuracy of training and test sets converged. This determination was further validated using 10-fold cross-validation. Next, using the determined value of k, various distance metrics, such as Euclidean, Manhattan and cosine were evaluated, along with the weighted k-NN using both the train and test splits. Finally the best performing distance metric was selected for evaluation on the test dataset.

2.3 Decision Tree

Since the Decision Tree is prone to overfitting, different proportions of train-test splits were used for evaluation. Additionally, the

Decision Tree's two different splitting criteria were utilized: Information Gain and Gini Coefficient.

2.4 ZeroR classifier

It is a simple classifier that predicts the most common label.

3. Results

Model	Best Accuracy
ZeroR	0.6142
Decision Tree	0.6312
K-NN	0.6724
Logistic Regression	0.7018

Table 1- model with their corresponding best accuracy

The table below (table 1) displays the best accuracies achieved by different models across different metrics. Below, the report discusses how each metric performed for each model.

3.1 Logistic Regression

This is by far the best-performing model among the other 3, with an accuracy of 0.7018, by using the holdout method. This is a one-vs-one logistic regression that uses the Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm.

	Accuracy	F1 score
Ovo (LBGFS)	0.702	0.650
Ovr (LBFGS)	0.692	0.630
Ovo (Newton-cg)	0.702	0.650

Table 2- Accuracy and f1 scores for different optimizing algorithms

	Holdout	10-CV
Ovo (LBGFS)	0.702	0.696
Ovr (LBFGS)	0.692	0.685
Ovo (Newton-cg)	0.702	0.696

Table 3- Accuracy of holdout and 10-fold cross-validation

From Table 2, it can be seen that one-vs-one logistic regression outperforms one-vs-rest logistic regression regardless of the optimization algorithm. As previously mentioned, 10-fold cross-validation was used to increase reliability the results from Table 3 further show the superiority of one-vs-one logistic regression over one-vs-rest logistic regression.

3.2 K-NN

K-NN achieved the second-highest accuracy among the four models with an accuracy of 0.672 using weighted K-NN and 10-fold cross-validation, with k =17.

	Train-Test	10-CV
Euclidean	0.652	0.660
Manhattan	0.650	0.657
cosine	0.665	0.660
weighted	0.669	0.672

Table 4- Accuracy of holdout and 10-fold cross-validation across different metrics

As evident from Table 4, weighted K-NN outperforms other metrics in both cross-validation and the holdout strategy. K-NN with cosine distance closely approaches the performance of weighted K-NN.

3.3 Decision Tree

Test Prop	Info_Gain	Gini
0.1	0.604	0.654
0.2	0.617	0.619
0.3	0.611	0.582
0.4	0.593	0.626
0.5	0.607	0.585
0.6	0.603	0.566
0.7	0.557	0.587
0.8	0.564	0.557
0.9	0.551	0.527

Table 5- Accuracy of different split proportion across different splitting criterion

As previously mentioned, the Decision tree was trained using different proportions of split for each splitting criterion. Surprisingly, the highest accuracy of the Decision Tree was achieved when the percentage of test cases was 10%. As seen from Table 5, the accuracy

value of 0.654 is achieved by the Gini Coefficient.

3.4 ZeroR

Class	Frequency
0	24
1	235
2	1839
3	777
4	129

Table 6- Class distribution of train dataset

ZeroR simply predicts the class with the highest frequency. From the class distribution in table 6, it is evident that Class 2 has the highest frequency. So, the model will predict any new instances as Class 2. Given that the proportion of class 2 is significantly higher than the rest, it is highly probable that a new unseen instance will belong to Class 2. As a result, zeroR has a very high test accuracy of 0.614.

4. Discussion and Critical Analysis

4.1 Logistic Regression

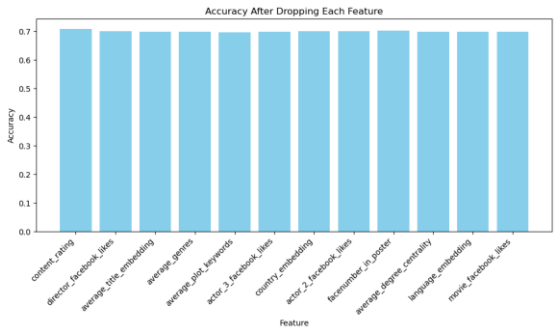

Figure 1- Accuracy of logistic model after dropping each feature

As long as feature selection is concerned, the accuracy of the logistic model does not change much, even after dropping the features with weights close to zero, which denotes the fact that redundant features are conveying the same information. Even though the accuracies change very slightly, the accuracy reaches its peak of 0.702 (from Table 2) after dropping 'facenumber in poster'.

	Before scaling	After scaling
Ovo (LBGFS)	0.657	0.702
Ovr (LBFGS)	0.670	0.692
Ovo (Newton-cg)	0.681	0.702

Table 7- Accuracy of model before and after scaling

The model benefits greatly from scaling. This could be due to the fact before scaling, features with larger magnitudes might have dominated the optimization process of finding the maximum number of iterations. Scaling the features likely helped the algorithm to converge to a more optimal solution. Feature scaling may have led to a better generalization as it performed better (0.644) on unseen data on Kaggle compared to the unscaled features (0.633).

Even though the accuracy improved after feature scaling and using an optimization algorithm, the accuracy is still relatively low.

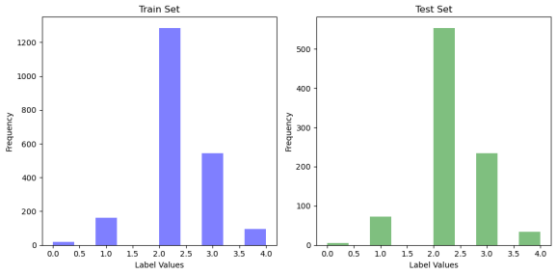

Figure 2- Class distribution of the training and test set

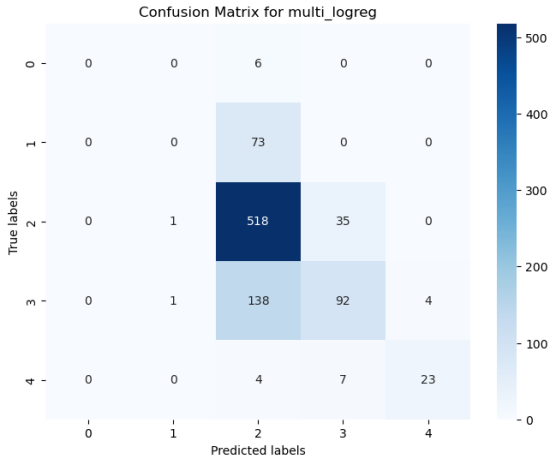

Figure 3- Confusion metric for one-vs-one logistic regression

Figure 2 illustrates the class distribution of the dataset which demonstrates that this is an imbalanced dataset. The frequency of class 2 is significantly higher compared to other

classes. Models tend to bias towards the majority class, which is Class 2 in this case, in an imbalanced dataset.

This bias is further proved by the confusion matrix in figure 3. The model is consistently predicting Classes 1 and 3 as class 2. The model is not learning to distinguish Classes 1 and 3 effectively, most likely due to Classes 1 and 3 having very few instances. It is evident that the model is having difficulty to capture the patterns from the minority classes [1].

4.2 K-NN

For K-NN tuning the hyperparameters is significantly important. K-NN is a "lazy learner". It does not capture the pattern, in turn, stores the train data in memory until it is required to use.

To find an optimal value of K, K-NN with several values of K was tested, and the value of K where the accuracies of train and test split converged was taken. This was further validated by doing 10-fold cross-validation, where the value of k found in the holdout strategy achieved the highest accuracy.

Even though weighted K-NN outperformed other distance metrics in both holdout and 10-fold cross-validation, the accuracies are very close to each other, indicating the distance metric has minimal effect on the model. The table below shows the Mean-Squared Error (MSE) for each metric, and it is evident that Weighted K-NN had the lowest MSE.

Metric	MSE
Euclidean	0.427
Manhattan	0.420
Cosine	0.400
Weighted	0.397

Table 8- MSE of each metric

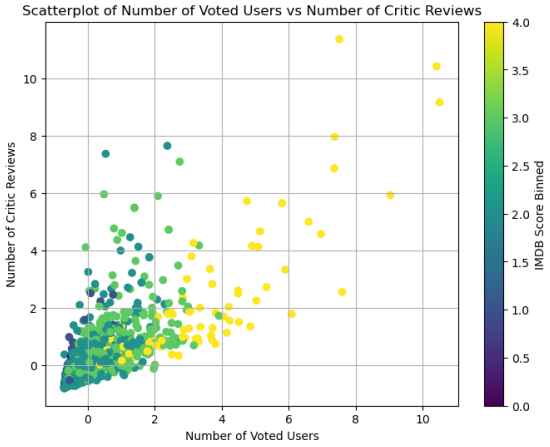

Figure 4- Scatterplot of Number of voted users vs Number of critic reviews

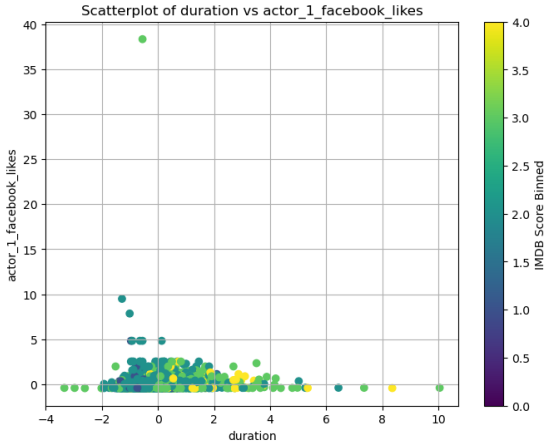

Figure 5- Scatterplot of actor 1 Facebook likes vs duration

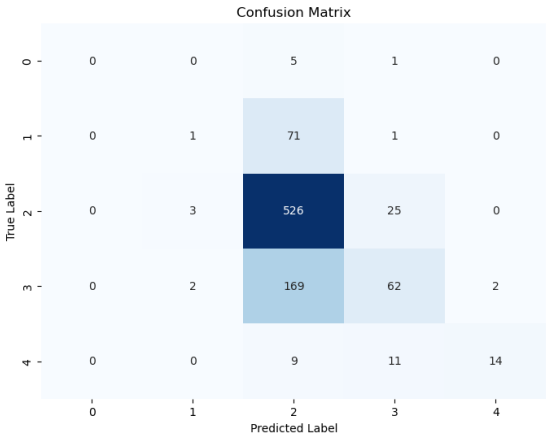

Figure 6- Confusion metric of K-NN classifier

Both Figures 4 and 5 show significant overlapping which demonstrates there are no clear decision boundaries between the class labels. This will significantly impact the ability of K-NN to make accurate predictions.

From Figure 6, it is evident that the K-NN model, like Logistic Regression, is inaccurately predicting

Classes 1 and 3 as Class 2. This is likely due to Class 2 dominating the distribution and since there is a significant overlap, the most common class around an unseen instance is Class 2.

4.3 Decision Tree

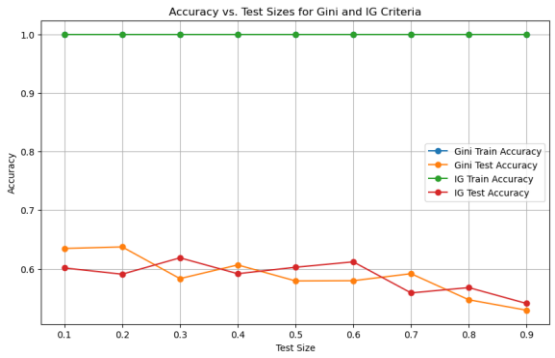

Figure 7- Accuracy Vs Test Sizes for Decision Tree

Figure 7 shows the Decision Tree achieves the maximum accuracy when the test size is just 10%. Moreover, as the number of test sizes increases, the accuracy of the Decision Tree decreases and performs worse than zeroR. This indicates the model is not very good at generalizing and is very closely fitted to the training data.

5. Conclusion

From these different models, it is evident that the imbalanced distribution of classes is presenting a challenge to predict the accurate class since the machine learning algorithms were designed to classify labels under the assumption that the labels in the dataset are equally distributed. Consequently, these models perform poorly in predicting minority classes [1]. Bootstrap sampling or oversampling may be used where more samples from the minority classes are picked to have an uniform distribution [2].

6. References

[1] Brownlee, J. (2019). A Gentle Introduction to Imbalanced Classification. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/what-is-imbalanced-classification/#:~:text=Imbalanced%20

[2] Or, D.B. (2024). Solving The Class Imbalance Problem. [online] MetaOr Artificial Intelligence. Available at: https://medium.com/metaor-artificial-intelligence/solving-the-class-imbalance-problem-58cb926b5a0f