School of Computer Science
University of Southampton Malaysia

Neu You Quan

15 December 2025

COMP3222 Machine Learning Coursework

Introduction

This project aims to predict IMDb movie ratings using a provided dataset containing 5,043 records. The goal is to classify ‘imdb scores’ into five different levels, from 1 to 5, respectively represent ‘Poor’, ‘Below Average’, ‘Average’, ‘Good’ and ‘Excellent’. The method employs a complete data processing pipeline which included removing duplicate data, imputing missing values, and encoding categorical features. The data were split into an 80% training set and a 20% test set, and resampling was done for training set only. A random forest estimator was used for feature selection and three supervised learning models used for comparison:Support Vector Classifier (SVC), Random Forest Classifier (RFC), Hist Gradient Boosting Classifier (HGB). The models were evaluated based on their ability to handle multi-class targets and class imbalance problems, and the best-performing model was selected for final hyperparameter tuning.

Methodology

Data Processing

The machine learning project had used a provided dataset containing 5043 rows of IMDB rating data which have 27 features columns and 1 target column name ‘imdb_score’. Following by initial preview, duplicate rows were dropped, which identified by same ‘movie_imdb_link’ to over using movie titles to strictly distinguish between remakes or alternate versions. A simple preliminary feature selection was performed after it, columns containing links and names related columns were dropped to prevent model overfitting to it. After it, the missing values were handled. ‘num_critic_for_reviews’ and ‘num_user_for_reviews’ were imputed by 0 as some null entries were confirmed to be 0 when verified through link and supported by minimum values over the column are 1. The remaining numerical columns were imputed by median values as ‘duration’, ‘gross’, ‘budget’, ‘title_year’ and ‘aspect_ratio’ were logically not to be 0 while 0 value already existed as minimum value in ‘facenumber_in_poster’ and Facebook like related columns. Object type columns were filled with the most frequent value (mode) to maintain overall data consistency.

Encoding techniques were applied after addressing missing value. ‘color’ and ‘language’ columns were transformed into binary features: ‘is_color’ and ‘is_english’, as the major class dominated the columns with 95.9% and 93.5% respectively. Similarly, the ‘country’ was transformed into two columns, ‘is_Uk’ and ‘is_USA’ as their representative of 8.8% and 75.5% respectively, while other countries were represented by 0 in both columns. As the columns ‘content_rating’, ‘genres’ and ‘plot_keywords’ contained multiple values in single cell, the values were split before applying one-hot encoding, total of 18, 26 and 8086 different values had resulted respectively. The columns ‘plot_keyword’ was dropped due to high cardinality, while the encoded ‘content_rating’ and ‘genres’ were concatenated into the dataset. The ‘imdb_score’ was replaced to ‘imdb_rating’ by converting score in interval of 2 (inclusive for ceiling) to categorizes from 1 to 5, with values <=2 mapped to 1, and values >8 mapped to 5.

Subsequently, outliers were capped using upper and lower bounds for all non-integer features columns (integer columns were those hot-encoding binary data preserving boolean nature). Year and count-based columns were cast to integers to prevent the generation of impossible floating-point values during synthesis of resampling. The final processed data frame was exported to a CSV file for future usage.

Pipeline

Features Selection and Pipeline

The model training workflows were implemented using a scikit-learn pipeline, with feature selection integrated directly into the process. The random state was set to be same to ensure the reproducibility, value of 42 was applied for this project. A Random Forest Classifier (RFC) was utilized for feature selection, followed by the training of three classifiers: Support Vector Classifier (SVC), RFC, and Hist Gradient Boosting Classifier (HGBC). During the feature selection stage, the RFC estimator was utilized, and a dynamic 'median' threshold was employed within the selection process. This ensured that only the top 50% of the most informative features contributed to RFC were retained.

Data Split

The data contain total of 4919 rows, which was partitioned into a training set of 80% and a testing set of 20% using a stratified split. This allocation is a standard strategy for datasets of this magnitude, it allocates the majority data for training to maximize the model's learning and a statistically significant portion for testing can avoid the luck only result, prevent the model from overfitting or underfitting. While stratify strategy allow the data split with the actual distribution of target classes, prevent the bias of split. The resampling was applied exclusively to the training set to mitigate heavy class imbalance. The three minor class are oversampling by ‘auto’ strategy with KN=3 to ensure robust synthetic sample generation as some classes is too small. Finally, the data was scaled by Min Max Scaler.

Model Selection

Support Vector Classifier (SVC)

SVC is the specific classification implementation of Support Vector Machines (SVM), a supervised machine learning algorithm which tends to separate the data with the best possible linear boundary plane. This approach is achieved by finding the widest gap between data points and boundary with the calculation of maximizes the margins among them. With ‘rbf’ kernel, the model is able to projects the data to a higher dimension, and then split the complex, non-linear data using a flat plane in that dimension. As the SVC is natively a binary class classifier, with, it uses a one-vs-one strategy to handle the multiple classes target, which train all models of classes pair combinations and take the highest voted class as result.

Random Forest Classifier (RFC)

RFC is a supervised ensemble model which implemented bootstrap aggregating(bagging) to form multiple decision trees and taking the results by voted. By sampling with replacement, slightly different subsets of data are created, and passed to multiple single decision tree for training. In decision tree, the node is built from the most informative feature, which is calculated based on Gini and entropy to become the node feature. However, the decision trees in random forest are forced to choose the most informative feature from a subset of features randomly selected every

time the tree reaches a node instead of from all features. The predictions from all trees are considered parallelly and apply majority voting to get the result. The default ‘n_estimator’ is 100, which mean 100 of trees are used, with ‘n_jobs’=-1, model will run trees parallelly to get the result faster.

Hist Gradient Boosting (HGB)

HGB is an evolution of gradient boosting, a supervised ensemble model that connects multiple weak learners (normally decision tree) and performs error focused optimization. Through the sequentially connection, each subsequent model learns from the errors (residuals) of the previous model, scaled by learning rate to improve training. The process repeats until the error stops decreasing or the maximum iterations are reached. In the HGB, the model does not process data as unique value like standard gradient boosting, it uses histograms to group continuous data into smaller number of bins, which significantly accelerates processing speed and regularizes data to reduce overfitting. The default loss function is set to ‘log_loss’, which calculates the logarithmic loss, while ‘max_iter’ (default 100) specifies the maximum number of training iterations allowed.

Selection Reason

The SVC was chosen for its robustness in high-dimensional environments and versatility in multi-class classification. Feature scaling was applied during preprocessing to ensure optimal convergence, and default hyperparameters were maintained, aside from the kernel specification. While the RFC was selected due to its inherent robustness to overfitting and high predictive accuracy. The model was initialized with default hyperparameters, utilizing parallel processing (n_jobs=-1) to optimize computational efficiency. Finally, the HGB was selected for its ability to process large datasets with superior speed while maintaining high accuracy. For this implementation, all hyperparameters were kept at their default values.

Evaluation Metrics

The model evaluation included the Confusion Matrix and aggregate metrics: accuracy, precision, recall, F1-score, and specificity. Confusion Matrix was plotted to visualize the distribution of the model's predictions against the actual class labels. As this was a multi-classes imbalanced classification problem, all aggregate metrics applied weighted averaging strategy. By this approach, the score for each class is weighted by its support to ensure its contribution to the final metric is proportional to its size in the dataset.

Cross Validation

The cross-validation was performed using an integrated pipeline that replicates the full preprocessing workflow (Resampling, Scaling, Feature Selection, and Model Training) within each fold. 5-fold Stratified Cross-Validation strategy was applied to maintains the original class distribution in each fold, which preventing the bias result cause by data splitting. The choice of 5-fold was primarily dictated by data constraints, the

minority class (Class '1') only have 7 instances, this set a strict upper limit on stratification. Besides that, 5 folds provide a balanced trade-off, preventing of not sufficiently validation sets. Furthermore, the shuffle was set to True to eliminate potential order dependency in the dataset, and parallel processing (n_jobs=-1) was utilized to increase the efficiency. The final performance of the cross-validation was evaluated using the mean value of same evaluation metrics (accuracy, precision, recall, F1-score, and specificity) among the 5 folds.

Hyperparameter Tuning

Grid Search approach was implemented to optimize the model's predictive power and prevent overfitting. The base model was wrapped in the same pipeline to ensure all processes was consistently applied. The search used 5-fold cross-validation (CV) and employed the weighted F1_weighted score as the primary metric for selection of the optimal model, which ensuring robust performance across all classes. The specific hyperparameters targeted for tuning varied by model, tailored to their unique characteristics and requirements.

Results and Interpretation

Model Confusion Matrix

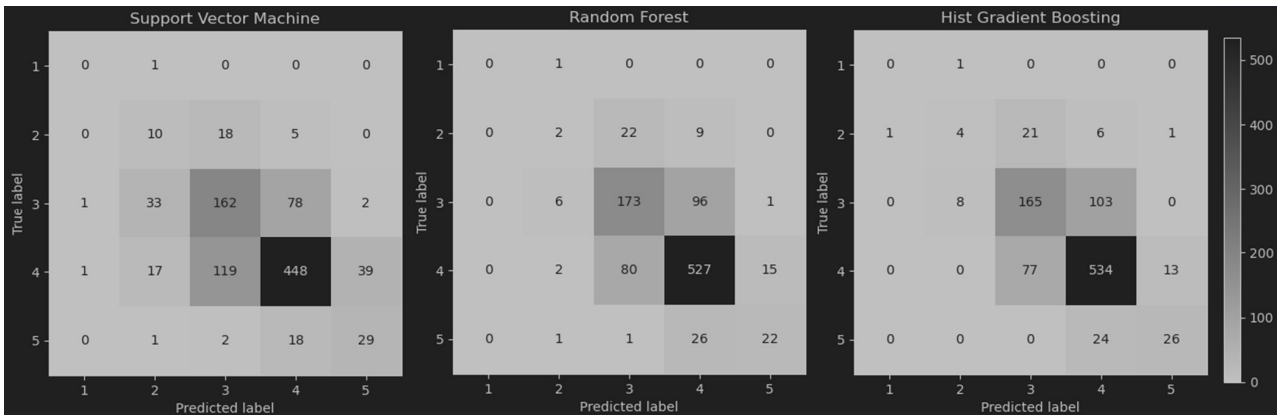

Figure 1. Confusion Matrix

Model Comparison

Table 1. Comparison of Models

		SVC	RFC	HGB
Model Architecture		Hyperplane-based / Distance-based	Parallel Ensemble (Bagging)	Sequential Ensemble (Boosting)
Strength		1. Effective for high dimension data 2. 2. Able to model complex, non-linear boundaries.	1. Robustness as many soft classifier 2. 2. No scaling needed	1. Optimize speed using histogram binning 2. 2. Natively able to handle missing values
Limitation		Feature scaling for convergence strictly requires	Prediction can be slow when numbers of trees increase	Easy to overfit on very small datasets
Test set	Accuracy	65.96%	73.58%	74.09%
	Precision	69.49%	71.97%	72.70%
	Recall	65.96%	73.58%	74.09%
	F1-score	67.35%	72.56%	73.15%
	Specificity	88.45%	89.26%	89.33%
Cross Validation	Accuracy	66.68%	73.00%	74.43%
	Precision	70.99%	72.26%	73.45%
	Recall	66.68%	73.00%	74.43%
	F1-score	68.28%	72.42%	73.70%
	Specificity	89.00%	89.31%	89.69%

Results and Interpretation

The confusion matrices in Figure 1 highlight the impact of class imbalance, resulted in a significant concentration of predictions in classes ‘3’ and ‘4’, while fewer predictions and consequently lower accuracy in the minority classes (‘1’ and ‘2’).

SVC model frequently misclassified instances from classes ‘3’ and ‘4’ as ‘1’ and ‘2’, which may cause by its mathematical interpretation attempts to fit decision boundaries but did not address the class imbalance issue. RFC model largely ignored the minority classes but boosting its accuracy by avoiding the associated errors paradoxically. This bias arises from the ensemble's voting mechanism, minority data is often absent from the bootstrapped subsets, preventing these classes from ever securing a majority vote during aggregation. In contrast, the HGB model demonstrates superior learning performance on minority class targets, predicting fewer but more accurate results. In contrast, the HGB model demonstrates superior learning on minority targets, yielding fewer but more accurate predictions. Notably, it avoids misclassifying class ‘4’ as a minority class, distinguishing it from the other models. This performance is likely driven by its error-focused gradient boosting strategy, which iteratively adjusts to define optimal prediction criteria for each specific class rather than being overwhelmed by the majority.

The confusion matrix for SVC displays a diffuse distribution, and RFC shows a distribution biased toward the majority; however, HGB exhibits a strong diagonal concentration, indicating it achieved the best overall performance among the three models. These visual findings are corroborated by the aggregate metrics (accuracy, precision, recall, F1-score, and specificity) in Table 1. HGB achieved the highest scores across all metrics, attaining 74.09% accuracy and a 73.15% F1-Score. This performance is significantly superior to SVC (65.96% accuracy, 67.35% F1-Score) and marginally better than the 73.58% accuracy and 72.56% F1-Score in RF.

Finally, HGB was selected as the final candidate to process hyperparameter tunning for further optimization due to its best performance over the evaluation metrics and high efficiency on large datasets. During the model tuning optimization, 5 specific hyperparameters were targeted, which included ‘learning rate’, ‘max iter’, ‘max depth’, ‘max leaf nodes’ and ‘l2 regularization’. The ‘learning rate’ controls the contribution of each tree to the ensemble model, while ‘max iter’ defines the maximum number of boosting iterations (trees) performed. The 'max depth’ and ‘max leaf nodes’ parameter limit the vertical growth and total terminal leaves of the single weak learners (decision trees) within the ensemble model respectively. Finally, ‘l2_regularization’ was included to preventing overfitting by penalize overly complex models.

The optimal HGB model identified via Grid Search having the following parameters: ‘learning rate’=, max iter’ = 200, 'max depth’ = 10, ‘max leaf nodes’= 20, and ‘l2_regularization’= 0.02. This configuration achieved accuracy of 74.59% and F1-score of 73.79%, marking a slightly improvement over the base model. The confusion metrics and aggregate results images were attached in Appendix A2. It is worth noting that the search was limited by number of values tested for each parameter as the processing time and computation power needed will increase significantly for a more thorough search (more values with smaller gap).

Summary

Machine learning pipeline was successfully implemented to classify movie ratings. A major challenge encountered was the class imbalance in the dataset, which were much of missing value and mitigated using stratified sampling and synthetic oversampling on the training data. The analysis of performance differences among the three models: SVC struggled with the data distribution, yielding the lowest accuracy (65.96%). RFC achieved better accuracy (73.58%) but biased its predictions toward majority classes. Hist Gradient Boosting (HGB) demonstrated the strongest performance, achieving the highest accuracy (74.09%) and effectively handling minority classes due to its error-focused learning strategy. Based on these results, HGB was chosen for optimization. Using Grid Search to tune parameters such as learning rate and tree depth, the final model achieved an improved accuracy of 74.59% and a weighted F1-score of 73.79%. The results conclude that the HGB model is the most robust and efficient choice for this specific dataset.

References

Appendix

A1

Jupyter notebook PDF link:
https://github.com/youquanneu/COMP3222_Coursework/blob/master/Report/Jupyter%20Notebook_PDF.pdf

GitHub link:

https://github.com/youquanneu/COMP3222_Coursework.git

A2

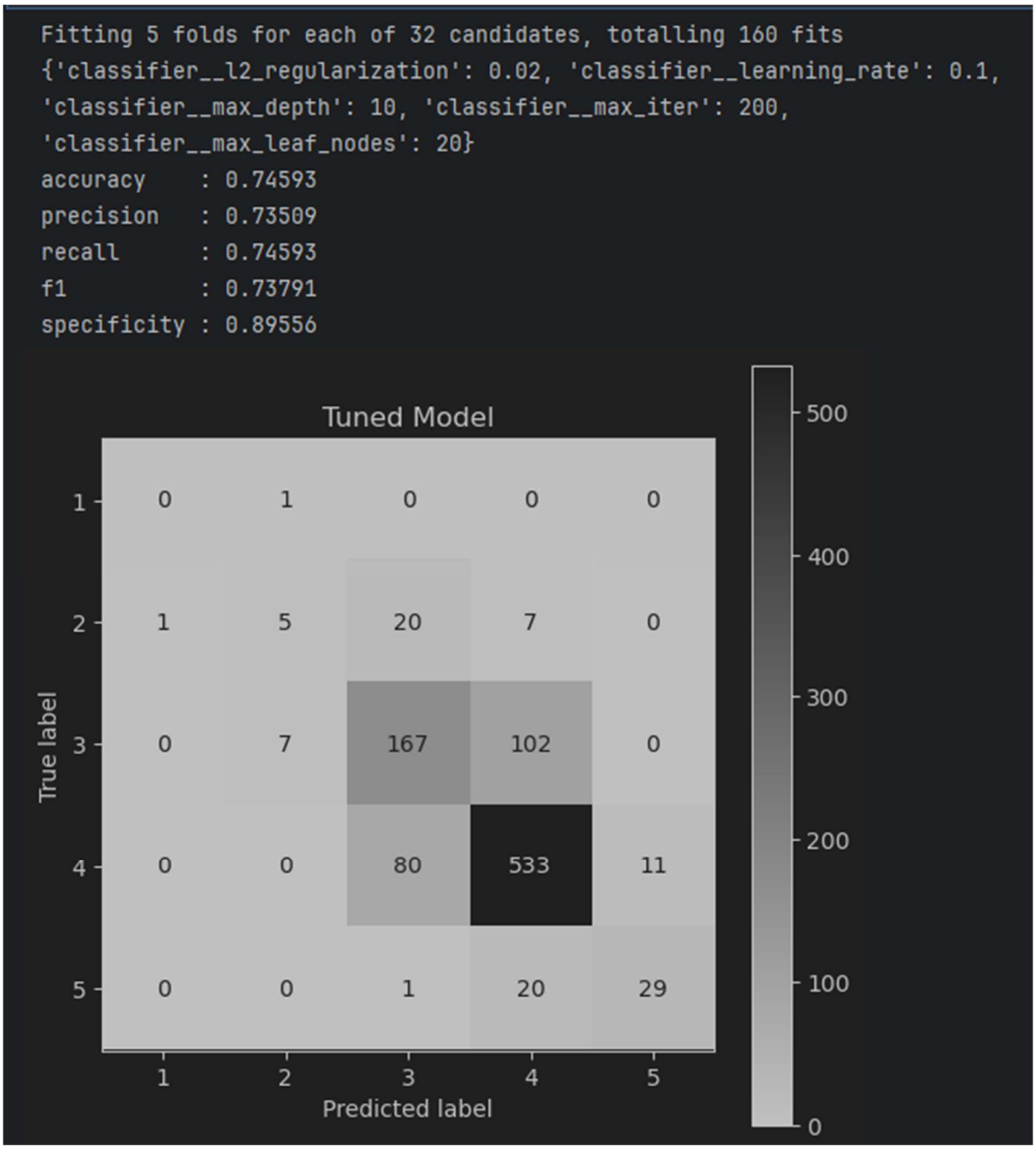

Figure 2. Result of Optimized HGB model

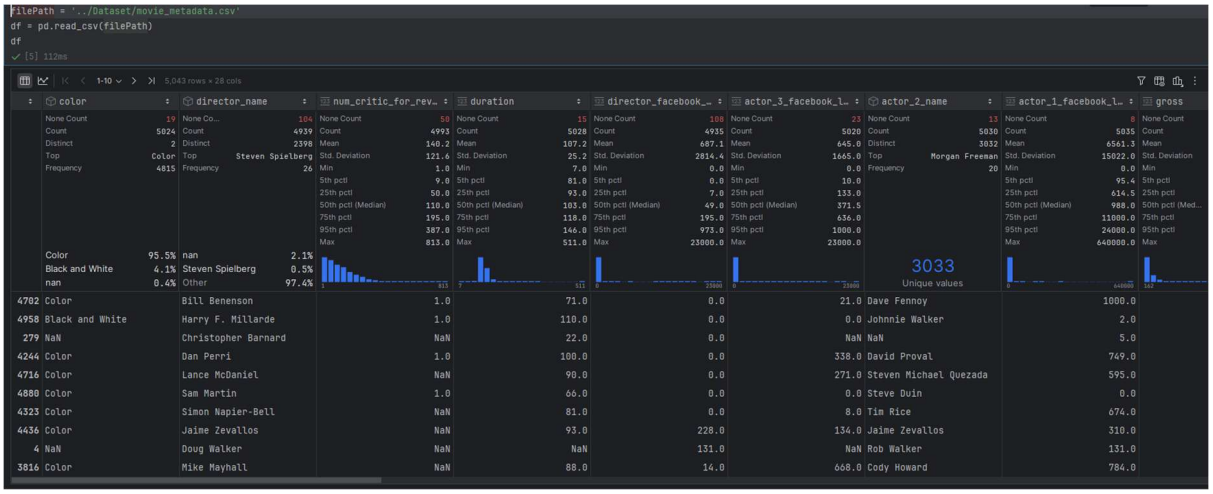

Figure 3. Inbuilt Data Analysis of PyCharm

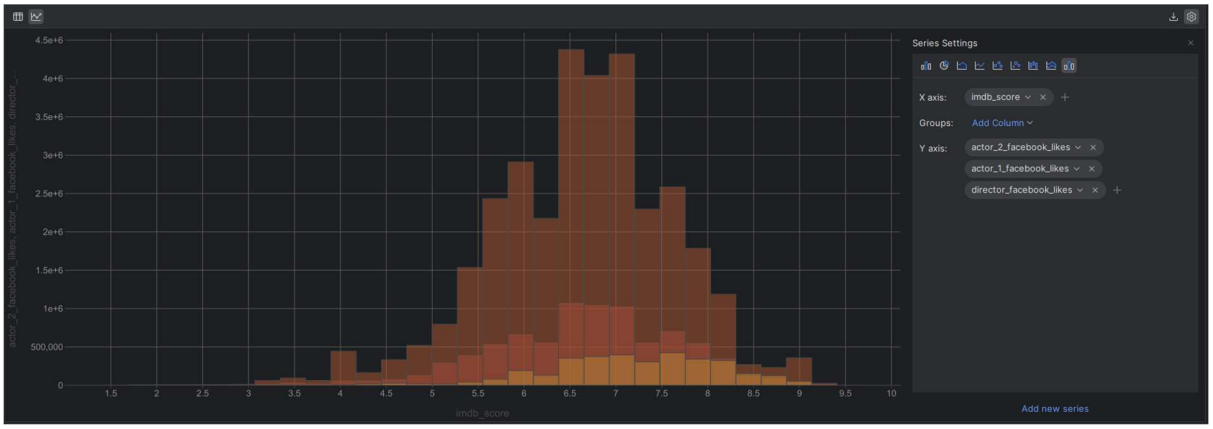

Figure 4. Inbuilt Graph Plotting of PyCharm