COMP30027 Report

Anonymous

1 Introduction

The vast array of books in today's world can make it challenging for readers to find new books that match their interest. Online platforms such as Goodreads[1] have emerged to help readers navigate to their interest, by providing a comprehensive database of books. However, with the number of books increasing, it is crucial to develop an accurate recommendation system to predict users' ratings. Machine learning models offer a promising solution to predict the rating of each book. This report focuses on the prediction of book ratings including preprocessing, modeling, fine-tuning and error analysis.

2 Background

To predict a book's rating, we acknowledge that the influential factors exhibit complex, non-linear relationships and are characterized by high dimensionality. Consequently, we've chosen to employ logistic regression among probabilistic learners, instead of the Naive Bayes classifier. This decision is because logistic regression doesn't require the assumption of feature independence. For other models, Support Vector Machines (SVMs), utilizing the kernel trick, enable us to operate in a high-dimensional feature space. This makes it possible to identify a separating hyperplane for the classes, regardless of their linear separability in the original feature space. We have also elected to use the Random Forest algorithm, given its robustness and tendency to outperform logistic regression (Couronné et al., 2018).

3 Method

3.1 Preprocessing

3.1.1 Feature Scaling

Feature scaling is used on numeric features with their ranges varied a lot. Scaling is of significant help in distance-based algorithms such as SVM. There are four numeric features in the dataset, PublishYear, PublishMonth, PublishDay and pagesNumber. After scaling, these features are centering around zero. Scaling these features can enhance the performance of distance-based algorithms and improve model interpretability.

3.1.2 Text Feature Representation

To identify the optimal method for text data representation, we explored the utilization of word embeddings via the GloVe txt "glove.6B.50d.txt" for the 'authors', 'description', and 'name' attributes. These embeddings are a result of the Global Vectors for Word Representation methodology proposed by Pennington, Socher, and Manning (Pennington et al., 2014).

As a part of our analysis, we deployed Principal Component Analysis (PCA) to visualize the feature spaces processed by both GloVe and Doc2Vec models. For Bag-of-Words (BOW) representations, we implemented Singular Value Decomposition (SVD) due to its efficacy with sparse matrices. Upon examination of the visualizations, it was apparent that the Doc2Vec model produced more distinct clusters for each feature compared to its counterparts. Given the superior performance of the Doc2Vec model and the inherent information loss associated with the BOW representation, we elected to use Doc2Vec for text feature representation in our subsequent analyses.

3.1.3 Categorical Feature Encoding

For categorical features such as 'Publisher' and 'Language', we implemented One-Hot Encoding. We chose to do so because one-hot encoding is an effective way to convert categorical data into a form that can be provided to machine learning algorithms to improve prediction performance. However, we opted to exclude the 'Language' column from the training dataset as it was sparse with only 25.4% non-null entries,

[1] https://www.goodreads.com

and our empirical observations indicated that its inclusion could potentially degrade the performance of our models.

3.1.4 Train Test Split

The dataset was divided into a training set and a validation set, with 80% of the data used for training and 20% reserved for validation. The training set and the validation set applied to each model.

3.2 Zero-R baseline

Zero R is a simple baseline that predicts the most frequent class in the dataset. In this case, we use Zero R to evaluate the performance of other models by comparing their accuracy to the accuracy of Zero R.

3.3 Multinomial Logistic Regression

We conducted an investigation into the Multinomial Logistic Regression (MLR) model, an extension of binary logistic regression. The MLR model includes a hyperparameter, denoted as 'C', which signifies the relationship between the features and targets. Given that our feature set is high-dimensional with numerous columns, we incorporated a penalty term into our model. This addition aims to prevent overfitting, thus enhancing the model's ability to generalize from our dataset. There exist additional techniques for Logistic Regression, including the One-Versus-Rest approach. However, given its relative inefficiency and tendency to yield higher standard errors (Agresti, 2013), we decided not to pursue this particular method.

3.4 Support Vector Machine

Support Vector Machines (SVM) function works by identifying a line or hyperplane that effectively separates distinct classes. We explored the one-versus-one SVM approach using three different kernels: linear, polynomial, and Radial Basis Function (RBF). All SVM models share the same hyperparameter C, which balances the trade-off between the hyperplane's margin and the number of support vectors. Comparing SVM with default C equals to 1, we found that SVM with RBF has the highest accuracy thus we decided to fine-tune the RBF-SVM.

3.5 Random Forest

The Random Forest, an ensemble method that combines multiple decision trees, is renowned for its robustness in handling noise within datasets, offering a significant advantage over less complex algorithms such as Logistic Regression (Couronné et al., 2018). This algorithm's efficacy is predominantly determined by two crucial hyperparameters: the quantity of decision trees (termed as 'number of trees') and the subset size of features utilized for constructing each tree (known as 'feature sub-sample size'). In our study, we fine-tuned those two hyperparameters to enhance the predictive accuracy of our Random Forest model.

3.6 Stacking

We investigated stacking as an ensemble technique to improve the performance of our best classifiers. We employed Random Forest and Support Vector Machine as our primary learning models, while Multinomial Logistic Regression was chosen as our ultimate meta-classifier. For each base classifier, the optimal hyperparameters were utilized to maximize their performance.

4 Result

4.1 Zero-R baseline

Zero-R baseline has an accuracy of 0.70378 on the test set and 0.71125 on our validation set.

4.2 Multinomial Logistic Regression

The results of the logistic regression with C equal to 1.0 are shown below.

	Precision	Recall	F1-Score
3.0	0.49	0.20	0.29
4.0	0.74	0.92	0.82
5.0	0.51	0.12	0.19

Table 1: Evaluation Metrics for MLR

After fine-tuning the hyperparameter C, the results are as follows:

	Precision	Recall	F1-Score
3.0	0.53	0.14	0.22
4.0	0.73	0.96	0.83
5.0	0.70	0.04	0.07

Table 2: Evaluation Metrics for MLR after fine-tuned

After fine-tuning, the model's precision has improved for classes 3.0 and 5.0, but the recall for these classes has decreased, meaning the model is missing more actual cases. This is also reflected in the lower F1-scores. The performance on class 4.0 has improved slightly.

These results suggest that while the model is getting better at making correct predictions, it's predicting class 3.0 and 5.0 less frequently.

4.3 Support Vector Machine

As anticipated, among the three variants of the Support Vector Machine (SVM), the Linear SVM demonstrates the lowest accuracy. The Radial Basis Function (RBF) SVM slightly outperforms the Polynomial (Poly) SVM. The respective confusion matrices can be found in Figure 1, Figure 2, and Figure 3.

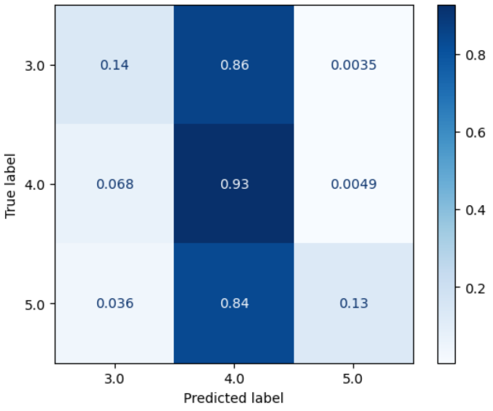

Figure 1: Confusion Matrix for Linear SVM

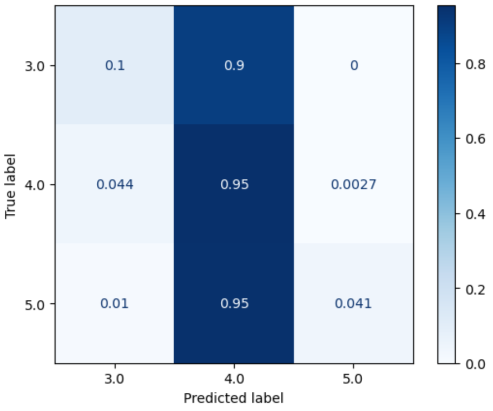

Figure 2: Confusion Matrix for Poly SVM

After using gridsearch with cross validation to fine-tuning the hyperparameter C for RBF-SVM within values 0.1, 1, 10, 100, 1000, we found that the best hyperparameter still equals 1.

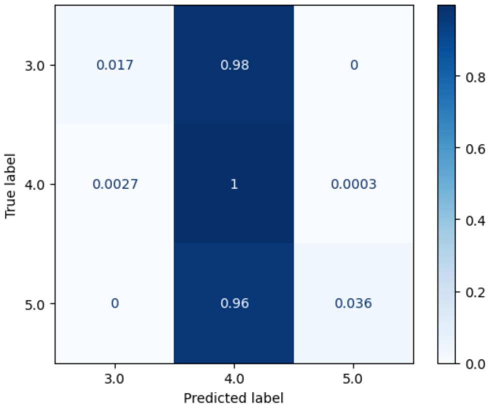

Figure 3: Confusion Matrix for RBF SVM

	Precision	Recall	F1-Score
3.0	0.68	0.02	0.03
4.0	0.71	1.00	0.83
5.0	0.88	0.04	0.07

Table 3: Evaluation Metrics for RBF-SVM

4.4 Random Forest

After fine-tuning, it is noteworthy that the model is biased towards class '4.0' and fails to classify '3.0' and '5.0'. The accuracy of Random Forest on test set is 0.70447, which is very close to the Zero-R baseline.

	Precision	Recall	F1-Score
3.0	1.00	0.14	0.22
4.0	0.71	1.00	0.83
5.0	0.00	0.00	0.00

Table 4: Evaluation Metrics for Random Forests

In addition, during the fine-tuning, the mean test score doesn't vary much with different hyperparameters, indicating that the model's performance is not sensitive to these particular hyperparameters over the ranges tested.

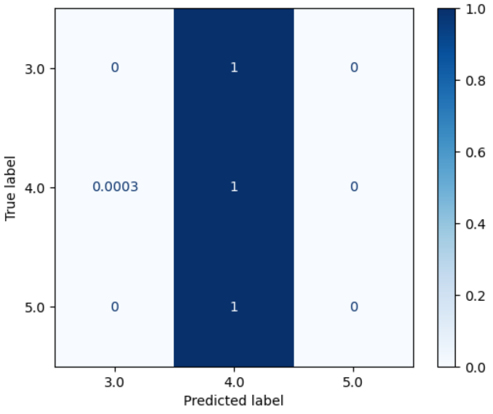

Figure 4: Confusion Matrix for Random Forest

	mean_fit_time	params	mean_score
0	9.798661	{sqrt, 100}	0.700434
1	19.322131	{sqrt, 200}	0.700976
2	29.063299	{sqrt, 300}	0.700867
3	38.558743	{sqrt, 400}	0.700759
4	47.643753	{sqrt, 500}	0.700759
5	4.567824	{log2, 100}	0.700813
6	8.917230	{log2, 200}	0.700759
7	13.235819	{log2, 300}	0.700705
8	17.554845	{log2, 400}	0.700705
9	21.926901	{log2, 500}	0.700705

Table 5: Hyperparameter tuning results for Random Forest

4.5 Stacking Model

After stacking, the model performance does not outperform any other models. The results can be found in Table 6.

	Precision	Recall	F1-Score
3.0	0.66	0.02	0.04
4.0	0.71	1.00	0.83
5.0	0.88	0.04	0.07

Table 6: Evaluation Metrics for Stacking Model

5 Analysis

Our results reveal that most models surpass the Zero-R baseline in terms of accuracy on our validation set, indicating that most models can to some degree use the features to effectively make decisions. However, the random forest model falls short of the baseline accuracy. Given its reputation as a robust classification tool, and considering previous studies where random forests outperformed logistic regression in 69% of datasets (Couronné et al., 2018), this outcome is unexpected.

One plausible explanation for this anomaly could be tied to our preprocessing strategy, where we used a one-hot encoder for the categorical data ”Publisher”, resulting in a sparse matrix. In such a matrix, a substantial number of entries are zeros. Logistic regression is less sensitive to this sparsity because it only uses the information from the features that are not zero to make predictions. Random forest, on the other hand, uses all of the features, including the zeros, to make predictions, which can make it challenging for the Random Forest to find optimal splits.

Another noteworthy observation is that the high overall accuracy of all models is primarily driven by their performance on class 4.0, while their predictions for class 3.0 and 5.0 are suboptimal, indicating that accuracy is not a good metric in this scenario. It can be revealed in the fine-tuning of logistic regression. After fine-tuning, even though the overall accuracy of MLR increases, the recall and F1-score for both class 3.0 and 5.0 decreases. Also, all other models exhibit a low recall and F1-score for class 3.0 and 5.0. This could likely be attributed to the bias in the dataset towards class 4.0. In the training set, test set, and our validation set, class 4.0 is the majority, leading to fewer examples from other classes for the models to learn from. The performance of random forests could be a reflection of that. With different settings of hyperparameters, the accuracy did not vary too much.

The stacking model, unfortunately, does not work as well as expected. Its performance, as revealed in the report (Table 3 and Table 6) , is nearly the same as the RBF-SVM. This suggests that all models are making similar decision boundaries between the classes. Consequently, the stacking model is unable to maximize the strengths and minimize the weaknesses in the diversity of the models, resulting in performance that mirrors its components.

Three variants of SVMs, as we expected, Poly SVM and RBF SVM have better performance than linear SVM. This suggests that the features within our dataset are not linearly separable.

A potential improvement we could make is to fine-tune the dimensionality of text features preprocessed by Doc2Vec. Post-preprocessing, we did not employ any techniques to reduce the dimensionality. The dimension of the feature space significantly impacts both the training time and the performance of each model.

6 Future Study

As the Large Language Model (LLM) trend takes center stage, we are curious about the developments in AutoML tools. To this end, we experimented with AutoGluon (Erickson et al., 2020). Surprisingly, its results outperformed all other submissions on Kaggle, highlighting its powerful capabilities.

However, while AutoML automates a range of processes such as preprocessing, model selection, and ensembling, it often results in complex models with limited interpretability. This complexity underscores a significant drawback in the context of understanding and explaining model decisions and behaviors.

Despite AutoML's impressive abilities, we firmly believe that it cannot entirely supplant human intervention. The human touch remains indispensable in interpreting results, posing the right questions, and making informed decisions based on the insights offered by these models.

7 Conclusion

In conclusion, with the exception of random forests, all our models surpass the performance of the Zero-R baseline. The underperformance of the random forest model is likely attributable to the dimensionality and the format of our training data. Interestingly, all the other models demonstrate remarkably similar performance levels, indicating that they are capturing similar patterns within the data. To solve this, we could resample the dataset to make it balanced. Future work could also investigate the use of dimensionality reduction techniques or explore other classification models to further enhance performance.

References

A. Agresti. 2013. *Categorical Data Analysis*. John Wiley & Sons, 3rd edition.

Raphael Couronné, Philipp Probst, and Anne-Laure Boulesteix. 2018. Random forest versus logistic regression: A large-scale benchmark experiment. *BMC Bioinformatics*, 19(1).

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. 2020. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.