Classification of book ratings

Anonymous

Word count: 2745

1 Introduction

Classification is a machine learning method that involves building a model to assign class labels to data instances based on their features. The model is a function $f : X \to C$, where X is the set of possible instances and C is a finite set, representing the possible class labels.

In the context of book publications, classification models can be used to predict the ratings of books, providing valuable insights into their potential popularity. This study aims to develop and evaluate several classifiers to predict the rating $r \in \{3, 4, 5\}$ of books.

The report is structured as follows: Section 2 describes the dataset, Section 3 discusses feature engineering, Section 4 elaborates on the application and evaluation of each classifier, Section 5 discusses the performance of each model and Section 6 draws final insights.

2 Data description

The dataset used for model training is sourced from GoodReads, a popular book-reviewing platform. Each instance has 9 features: *Name, Authors, PublishYear, PublishMonth, PublishDay, Publisher, Language, pagesNumber, Description*, and a categorical label *rating_label* $r \in \{3, 4, 5\}$. The training set contains 23063 labelled instances, and the test set contains 5766 unlabelled instances. The distribution of classes in the training set is shown in table 1. Assuming that the training set contains randomly-sampled instances, this suggests that the number of books where $r = 4$ is far more than books with $r = 3$, which is in turn a lot more than books with $r = 5$.

Label	3.0	4.0	5.0
Number of instances	5864	16208	991
Percentage	25.4%	70.3%	4.3%

Table 1: Distribution of labels

3 Methodology

3.1 Feature engineering and selection

We conducted feature engineering, aiming to convert all features into numerical format for effective training of the classification models.

Firstly, we transformed features *PublishYear, PublishMonth,* and *PublishDay* from the original dataset into a numerical representation via the concatenation of the three features. For example, the date 2018/2/12 was converted to the value 20180212. This transformation preserved the relative ordering of the dates, enabling the comparison of publishing time directly.

Next, we applied one-hot encoding to the *Language* feature, creating of 19 binary variables representing different languages such as *eng, spa, fra.* This encoding approach was chosen to turn the categorical feature into multiple binary features (0 or 1) that can be processed easily, due to its numeric nature.

Both doc2vec and count-vectorisation are techniques used to encode text features into multiple numeric features. Count-vectorisation represents the occurrence of each word in a text as a vector, where each column (feature) in the vector corresponds to a word in the corpus. As a result, the number of features generated by count-vectorization can be quite large, depending on the size of the vocabulary. Doc2vec, on the other hand, produces fixed-size vectors that capture the overall semantics of the text. These vectors abstract the occurrence of individual words in the text and in turn provide a more compact representation compared to count-vectors. To capture the important semantic meaning of features *Description* and *Name*, we chose to use doc2vec representations over count vectors, which also has less complexity and overfitting-risks due to having less features. On the other hand, for features *Authors* and *Publishers*, we chose count-vectorisation. We expect that books with similar authors or publishers will have similar ratings, and by using a count vector, we can capture the specific names of authors and publishers.

3.2 Preprocessing

In the training set, there are 148 instances without the *Publisher* feature and 17,202 instances without the *Language* feature. One approach to address missing values for *Language* is by imputing them with the mode value, ”eng” (English). However, examining the label distributions for instances with missing *Language* and those with non-missing values (Table 2) revealed distinct differences between them. Imputing missing *Language* values with the mode could potentially introduce bias into the dataset. Hence, we decided to impute missing *Language* as “Unknown”. Missing *Publisher* values were also imputed as “Unknown”.

Language	3.0	4.0	5.0
missing	27%	68%	5%
not missing	19%	78%	3%
eng	19%	78%	3%

Table 2: Percentage distribution of labels, based on if *Language* is missing

To create the final set of features for our analysis, we concatenated multiple features into a single sparse matrix. The selected features included *pageNumber*, the doc2vec representations of *Description* and *Name*, count vectorisation of *Publisher* and *Authors*, the numeric time feature, and the one-hot encoded representation of *Language*. This resulted in a total of 17,146 features.

In order to make distance-based learners such as SVM and KNN more performant, the numeric features are scaled using StandardScaler from *sklearn*. This way, features have a mean of 0 and variance of 1 while also preserving the original shape of the data distribution.

4 Model training and evaluation

4.1 Training Process

The training process involves preprocessing the provided training set, followed by an 80-20 split into a training and validation set (we will refer to “training set” as the training set after the split). For each classification model, we used grid search and stratified 5-fold cross-validation on the training set to find a set of hyperparameters that maximizes model accuracy. Stratification is chosen to preserve the original data distribution within each fold, reducing model and evaluation bias, while cross-validation helps control variance in our evaluation and prevents overfitting, ensuring that our hyperparameters are optimal. Finally, the models are trained on the training set, and their performance is evaluated by predicting labels on the validation set and reporting the relevant metrics.

4.2 Evaluation metric

We focus on the following metrics to assess our models:

- Accuracy: $(TP+TN)/(TP+TN+FP+FN)$. This assesses the overall performance of our model.
- Precision: $TP/(TP+FP)$. This gives an indication whether the classifier is being over-optimistic.
- Recall: $TP/(TP+FN)$. This metric will be used to see if the model has bias towards the majority class (4).

4.3 0R classifier

The 0R classifier serves as a baseline model, predicting the majority class (rating 4.0) which represents 70.28% of the training instances. The model achieves an accuracy of 70.02% on the validation set, but class 4 has high false-positives and classes 3 and 5 have a recall of 0, indicating high bias.

4.4 Nearest Neighbour classifier

The k-Nearest Neighbour (KNN) classifier predicts an instance’s class based on the votes of its k-closest neighbouring instances in the training data. The voting weights can either be uniform or based on the distance to the instance being predicted.

We performed grid search over parameters $weights \in \{\text{“}distance\text{”}, \text{“}uniform\text{”}\}$, $n\_neighbours \in \{5, 10, 20, 40, 80\}$, acquiring the highest accuracy model with parameters $n\_neighbours = 80$, $weights = \text{“}distance\text{”}$.

Label	Precision	Recall	F1-Score
3.0	0.00	0.00	0.00
4.0	0.70	1.00	0.82
5.0	0.00	0.00	0.00
Accuracy			0.70
Macro Avg	0.23	0.33	0.27
Weighted Avg	0.49	0.70	0.58

Table 3: Metrics for KNN predicting validation set

From 3, it can be observed that the model's performance is identical to the 0R baseline. It achieves an accuracy of 70% and only predicts label 4. Essentially, this KNN model is significantly under-fit and shows a strong bias towards the majority class, 4.

During cross-validation, as $n\_neighbours$ decreases, accuracy decreases and variance across folds increases. This reduction in $n\_neighbours$ also decreases the bias towards label 4 and allows for occasional predictions of labels 3 and 5.

4.5 Logistic regression

The Logistic regression classifier predicts labels using a logistic function to map input features to the predicted probabilities of it being in a class. It has hyperparameters C, which controls the regularisation strength and balances model complexity, and *max_iter*, which determines the maximum number of iterations allowed for the optimisation algorithm to converge.

We performed grid search over parameters $C \in \{0.1, 1, 10, 100\}, max\_iter \in \{500, 1000, 2000\}$, acquiring the highest accuracy model with parameters $C = 0.1, max\_iter = 500$.

Label	Precision	Recall	F1-Score
3.0	0.58	0.18	0.27
4.0	0.73	0.95	0.82
5.0	0.60	0.09	0.15
Accuracy			0.71
Macro Avg	0.64	0.40	0.42
Weighted Avg	0.68	0.71	0.65

Table 4: Metrics for logistic regression predicting validation set

Actual \Predicted	3.0	4.0	5.0
3.0	206	969	2
4.0	146	3074	10
5.0	1	187	18

Table 5: Confusion matrix for logistic regression

Based on Table 4, the observed accuracy of the model is 71%, surpassing that of the 0R model. The model also demonstrates reasonable precision in predicting the labels. However, it exhibits noticeable bias towards label 4, as it is predicted 91% of the time despite accounting for only 70% of the population in the validation set. Consequently, this bias leads to a significant number of false negatives when predicting classes 3 and 5, resulting in low recall values for these classes.

4.6 Support vector machine

The Support Vector Machine (SVM) classifier uses a kernel function to map instances into a higher-dimensional space. Its objective is to find an optimal hyperplane that effectively separates data instances into their respective classes. SVM has hyperparameters C and *gamma* that control the trade-off between maximizing margin and training errors, both of which impact the model's performance and ability to generalise.

We performed grid search on hyperparameters $C \in \{0.1, 1, 2, 4, 8\}$, $kernel \in \{\text{“}rbf\text{”}, \text{“}poly\text{”}\}$ and $gamma \in \{\text{“}auto\text{”}, \text{“}scale\text{”}\}$. The best selected model has $C = 2, kernel = \text{“}rbf\text{”}, gamma = \text{“}scale\text{”}$. As shown in Table 6, this model achieves an accuracy of 71%, which is higher than the 0R model. However, similar to the logistic regression classifier, it exhibits a bias towards label 4, resulting in low recall values for labels 3 and 5. While we achieve perfect precision in predicting label 5, there is a significant number of false negatives, adversely affecting the overall accuracy of the model.

Label	Precision	Recall	F1-Score
3.0	0.58	0.12	0.19
4.0	0.72	0.97	0.83
5.0	1	0.05	0.10
Accuracy			0.71
Macro Avg	0.77	0.38	0.37
Weighted Avg	0.69	0.71	0.63

Table 6: Metrics for SVM predicting validation set

Actual \Predicted	3.0	4.0	5.0
3.0	136	1044	0
4.0	97	3142	0
5.0	3	181	10

Table 7: Confusion matrix for SVM

For a SVM with $kernel = \text{“}rbf\text{”}, gamma = \text{“}scale\text{”}$, as C is increased (making SVM less tolerant to training error), the recall for label 3, 5 increases, but their precision decreases. Meanwhile, the precision for label 4 increases and its recall decreases.

4.7 Stacking Model

The stacking ensemble model combines many base classifiers and uses a meta-classifier to make the final prediction. Our stacking model includes six base classifiers:

1. **0R**
2. **KNN**: uses 2000 features, $n\_neighbours = 9, weights = \text{“}distance\text{”}$
3. **SVM**: uses all features, $kernel = \text{“}rbf\text{”}, gamma = \text{“}scale\text{”}, C = 2$
4. **SVM, balanced**: uses 6000 features, $kernel = \text{“}rbf\text{”}, gamma = \text{“}scale\text{”}, C = 4, class\_weights = \text{“}balanced\text{”}$
5. **Logistic regression**: uses all features, $max\_iter = 500, C = 0.1$
6. **Logistic regression, balanced**: uses 6000 features, $max\_iter = 2000, C = 12, class\_weights = \text{“}balanced\text{”}$

Model 2,4,6 are not yet introduced and will be covered in the discussion section. They require feature selection using mutual information as the metric.

Label	Precision	Recall	F1-Score
3.0	0.60	0.23	0.34
4.0	0.74	0.94	0.83
5.0	0.65	0.20	0.31
Accuracy			0.73
Macro Avg	0.66	0.46	0.49
Weighted Avg	0.70	0.73	0.68

Table 8: Metrics for Stacking Model predicting validation set

From table 8, we observe that the ensemble model demonstrates the highest accuracy among all the models, reaching 73%. Additionally, when considering the weighted average of precision and recall, this ensemble model achieves the highest performance compared to the previously discussed models. Although there is still a bias towards label 4, with the model predicting it approximately 89% of the time, the number of minority labels misclassified as 4 is comparatively lower than that of the previous models as seen in table 9. Notably, the ensemble model achieves better classification results for instances with labels 3 and 5 in terms of having a higher F1 score.

Actual \Predicted	3.0	4.0	5.0
3.0	274	902	1
4.0	179	3029	22
5.0	2	162	42

Table 9: Confusion matrix for Stacking Model

5 Discussion

5.1 Nearest Neighbour classifier

The KNN classifier was selected under the hypothesis that books with similar features would belong to the same class. However, as indicated in Table 3, the model performs poorly and deteriorates to the 0R baseline, where all instances are classified as the majority class (4).

For instances with label 5 in the validation set, we examined their neighbours when classifying them using our model. On average, approximately four neighbours among their 80 nearest neighbours have label 5. The nearest training instance with label 5, on median, ranked as the 6th nearest point overall. These findings suggest that, in our current model, instances with a rating of 5 exhibit significant dissimilarities in their features, leading to the domination of the majority class during the voting process. This explains the model's failure to classify instances with label 5. The underlying cause of this error may be attributed to the presence of noisy features and curse of dimensionality, where in high-dimensional spaces, the notion of distance becomes less reliable, since instances are more sparsely distributed.

To address this issue, we performed feature selection, selecting k features that have the highest mutual information with the class label. To find the optimal k, we experimented with different feature sizes and ran a grid search with cross-validation over the parameter *n_neighbours* $\in \{1, 3, 5, 7, 9\}$. Since we want to prevent class 4 from dominating the voting process, we use small *n_neighbours* values to prevent underfitting. The best model by accuracy has $k = 2000$ features and *n_neighbours* $= 9$, with metrics shown in table 10.

Label	Precision	Recall	F1-Score
3.0	0.34	0.11	0.17
4.0	0.71	0.93	0.80
5.0	0.50	0.02	0.04
Accuracy			0.67
Macro Avg	0.51	0.35	0.34
Weighted Avg	0.60	0.67	0.60

Table 10: Metrics for Feature-Selected KNN

While the overall accuracy decreased slightly in comparison with table 3, predictions were actually made for minority classes 3.0 and 5.0. The smaller *n_neighbours* values significantly increased precision, recall for these classes, while the recall for class 4 decreased. Given the limited improvement in the overall performance, it is worth considering the incorporation of new features or exploring alternative algorithms, as KNN may not be the most suitable approach for this particular problem.

5.2 Logistic regression

Logistic regression was chosen as it is an interpretable and commonly used classifier for multinomial classification. By examining the coefficients of the features in our model, the following observations were made:

- "audio" has the highest absolute coefficient value when predicting labels 3 and 4. It has a coefficient of -0.68 for predicting label 3 and 0.64 for label 4. This word is found in the *Publishers* feature, suggesting that instances containing this term are more likely to be rated as class 4 rather than class 3.
- "anonymous" has the highest absolute coefficient value when predicting label 5, with a coefficient of 0.82. This word appears in the *Authors* feature, indicating that books with anonymous authors are more likely to be rated as class 5.

Label	3	4	5
Number of instances	1	13	35

Table 11: Distribution of label in training set when the *Authors* is "anonymous"

Based on table 5, the model misclassified many instances from classes 3 and 5 as class 4, indicating a bias towards the majority class. To address this error, we used the "balanced" option for the *class_weight* parameter, giving higher weight to classes 3 and 5. Additionally, we performed feature selection by choosing k features with the highest mutual information with the class label. This aimed to reduce noisy features and improve model performance. Through experimentation and grid search with cross-validation, we determined that selecting $k = 6000$ features, along with parameters $C = 12$ and *max_iter* $= 2000$, yielded the best accuracy.

Label	Precision	Recall	F1-Score
3.0	0.52	0.49	0.51
4.0	0.79	0.79	0.79
5.0	0.33	0.40	0.36
Accuracy			0.70
Macro Avg	0.55	0.56	0.55
Weighted Avg	0.70	0.70	0.70

Table 12: Metrics for balanced logistic regression

Table 12 shows the metrics for the new logistic regression model. Overall, we see a slight decrease in accuracy and a decrease in the macro average of precision, but recall for classes 3 and 5 greatly increased, indicating fewer false negatives and less bias. However, as our C parameter increases from 0.01 to 12, the strength of the regularisation is reduced and this may lead to overfitting. As we see from the two learning curves in figure 1 and 2, the gap between the training and validation curve for our new model with $C = 12$ is significantly larger compared to our old model, when $C = 0.1$. This suggests that our new logistic regression performs well on the training instances but fails to generalise to unseen samples, a sign of possible overfitting.

5.3 SVM

The SVM was chosen as it can handle complex decision boundaries between classes. It suffers from the same problem as the logistic regression classifier, where there are too many false-negatives for labels 3 and 5, resulting in high bias as seen from table 7.

We find that our model contains 16319 support vectors, while the training set only has

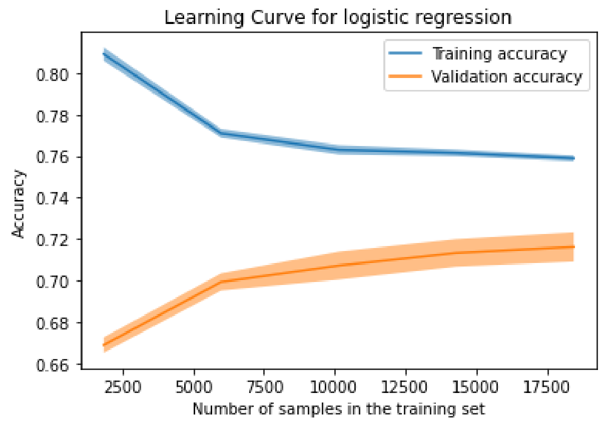

Figure 1: Learning curve for logistic regression, $C = 0.1$

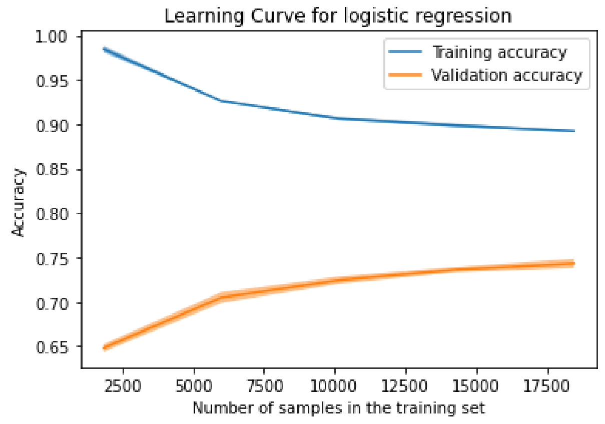

Figure 2: Learning curve for logistic regression, $C = 12$

18450 instances. The large number of support vectors indicated that our model has a large variance, as the margin is highly dependent on the training instances. We specifically looked at the distance of validation instances from the decision boundary of the 'one-vs-rest' SVM when the 'one' is class 5. The top 10 furthest instances from the decision boundary are all correctly classified with label 5, as seen from the confusion matrix in table 7. Their distance to the decision boundary is ≈ 2.2. However, the next 5 furthest instances are being classified as label 4, despite being actually of label 5, with their distance to the decision boundary ≈ 1.1. This sharp drop in distance resulted in the misclassification of label 5 instances. To address the high variance, we will increase C to have a smaller margin, and perform feature selection to have more informative features. Both will help to reduce the number of support vectors. To reduce bias, we will set the *class_weight* parameter to "balanced", so class 3 and 5 gets more weighting and potentially decrease the error caused by false negatives.

Label	Precision	Recall	F1-Score
3.0	0.44	0.42	0.43
4.0	0.76	0.80	0.78
5.0	0.43	0.15	0.23
Accuracy			0.67
Macro Avg	0.54	0.46	0.48
Weighted Avg	0.66	0.67	0.67

Table 13: Metrics for balanced SVM

Table 13 shows the new SVM after selecting 6000 features with the highest mutual information score, making $class\_weight = \text{"}balanced\text{"}, kernel = \text{"}rbf\text{"}$ and $C = 4$. Overall, this improves classifying label 3 the most, with the precision and recall both having a significant increase. The recall of label 5 tripled, however its precision decreased. The overall accuracy decreased by 4%, indicating the trade-off of having a higher macro average of recall versus accuracy. However, the new SVM did not end up reducing the number of support vectors, as it actually increased to 17056. This indicated that the current decision boundary is too complex, and we may need to engineer new features to fix this.

5.4 Stacking Model

The stacking model was chosen as it should perform as well as or better than its best base classifier.

Like its base classifiers, the model is biased in misclassifying most minority class instances as the majority class (4). This is likely because the base classifiers lack diversity in algorithm and fields of error; two SVM and logistic regression models are used, and most individual classifiers favors class 4. Hence, the ensemble model continued to carry the same biases. To address this error, we need to have a more diverse range of base classifiers, such as adding a Decision Tree and Naive Bayes model.

6 Conclusion

In conclusion, our analysis demonstrated that the stacking ensemble classifier outperformed other classifiers in terms of accuracy, weighted precision and recall. All trained models showed a bias towards the majority label 4.0, signaling that there is too much noise in the current feature set for the minority classes to be well captured.

For future improvements, new feature engineering methods can be used to reduce noise and extract more relevant information. Oversampling techniques can also be used to increase the representation of minority classes. To improve the performance of the stacking classifier, a more diverse range of base classifiers can be used.