Project 2: Book Rating Prediction

Anonymous

May 19, 2023

1 Introduction

The goal of this project is to produce and critique multiple machine learning models that have the goal of predicting the rating (3, 4, or 5) of books on Goodreads, based on a training data set that has been provided.

In machine learning models, the performance is generally measured through the accuracy of the model, that is, how many instances of a test set are predicted correctly. However, how the performance of a model should be evaluated depends heavily on the circumstantial data, and is at the author's discretion. In the Goodreads data set, a major problem that is encountered is the class imbalance. About 70% of the data is of class label '4.0', which means if we only measure the model's performance on accuracy we can create a model with reasonable performance (about 0.7 accuracy) without gaining any significant information. Therefore throughout this project, as well as focusing on accuracy, the performance of all our models will be measured through their F-score, and any binary models will use Area under the ROC Curve (AUC), which are commonly used in academia [2]. As well as this, the partition of the test data predicted as class label '4.0' is important, to ensure we are not over-predicting the major class.

Feature selection and feature reduction are another, albeit minor, complication that needs to be overcome for each model.

There are three models explored and analyses in this project. Model A utilises three binary K-Nearest neighbour algorithms with Chi-squared feature reduction, Model B is a Deep Neural Network (DNN) with Random over/under sampling and/or thresholding, and Model C is a mixture of what we believe to be the best ideas from both, involving three binary regression DNNs and stacking using a classification neural network. We also utilise ROS/RUS and Thresholding, as used in Model B.

2 Methodology

2.1 Model A

Model A uses three K-Nearest Neighbour (K-NN) algorithms as well as a classifier, which is used in the stacking of the models. A select number of numerical features of the training data are chosen in order to reduce the dimensionality of the training set. Each of the K-NNs are binary, that is, they try to predict the class label of all data points between only two of the labels. In this data-set, that means we have three models, respectively predicting between classes '3.0' and '4.0', '4.0' and '5.0', and '3.0' and '5.0'. Then a multi-layer perception classifier with a hidden layer of 4 is utilised to predict the class label based of these three K-NN prediction.

2.2 Model B

Model B is a deep neural network (DNN) with Random over-sampling then Thresholding applied during the fitting of the model. From the given data, only the pre-processed description doc2vec features are used, which is 100 numerical features. As neural networks do well at not using irrelevant features [4], no feature reduction is performed prior to training. The model, that is sequentially built using TensorFlow, has 4 hidden layers of decreasing size, all using the activation function Leaky ReLU with dropout after each layer, in order to prevent over-fitting. Then, the multi-class output of length three has the SoftMax taken. The model is then trained on 80% of the training data.

2.3 Model C

Model C is a mixture of ideas from Model A and Model B. The model again has three sub-models, however, each individual sub-model is a linear neural network. Then, as in Model A, a stacking model is used to classify each instance.

The sub-models are three binary regression DNNs across the one-vs-one models. We use ROS/RUS then thresholding in the sub-models, since as was learnt in Model B, this can help with class imbalance. Furthermore, the output to these models in input into a linear neural network that predicts each instance. All models were trained on 80% of the original training data, as was done in Model B.

3 Results

Model	Accuracy	F-score	Major class
Model A	0.582	0.584	0.710
Model B	0.667	0.639	0.819
Model C	0.668	0.643	0.807

Table 1: Model Comparison

In all the models, we can see that the major class, labelled '4.0' is over-predicted. Due to the nature of the training data, this is something that cannot be mitigated without compromising the accuracy of the model.

For example, Model B predicts that 81.9% of the data should be of the major class (Table 1), which over-estimates the prior distribution of the training data, of which about 70% is the major class. Conversely, Model A does a good job at predicting close to the prior distribution, at 71.0%, but its Accuracy and F-score are both significantly lower (Table 1). Model C does a better job than Model B while also improving the F-score (0.643 and 0.639, respectively) of the predictions, leading to a better model.

Looking at the confusion matrices of the models (Figure 1, we can see Model A is the best at predicting class label '5.0', but suffers predicting the other two labels. Models B and C predict '3.0' more correctly, but still over-predict the main class, however we think that Model C finds the best balance in accuracy and F-score.

4 Discussion

4.1 Model A

Weighting Algorithm in Model	Accuracy	F-score	Major class
Uniform	0.676	0.606	0.913
Distance	0.677	0.607	0.913
Inverse class label	0.582	0.584	0.710

Table 2: K-NN weighting algorithm comparison

The numerical features in the training data were all scaled to between 0 and 1, with the best 42 chosen through a Chi-squared method. The number

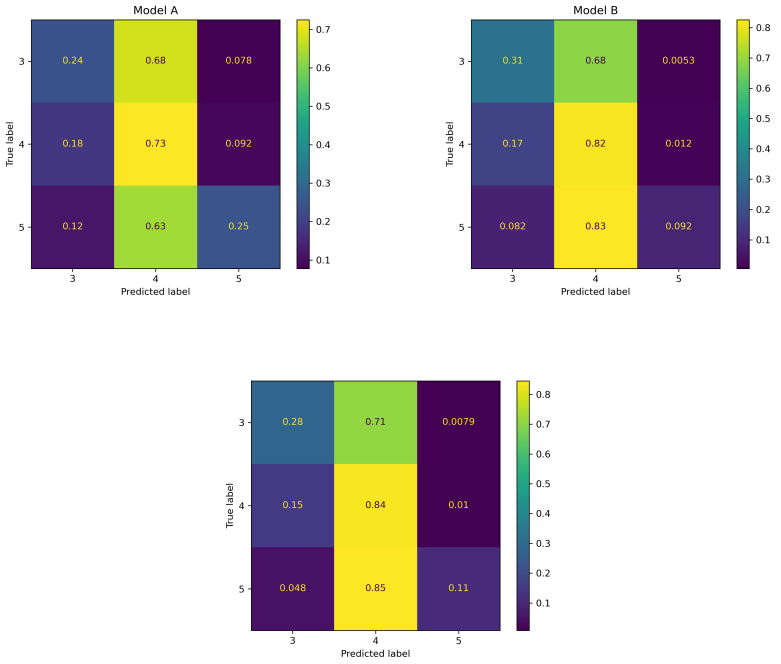

Figure 1: Confusion Matrices for Models

of features selected was empirically found to produce models with the highest F-score. In Figure 2, we can see the number of features used in a baseline K-NN against the F-score of the output produced, with the maximum being at 42, as we use.

Model A utilises the K-Nearest Neighbour algorithm, which has been proved to be a classification tool with good scalability and performance, but suffers from class imbalance, due to K-NN not taking into account the nature of the data around the prediction point [1]. To mitigate this, each of the three binary K-NN's distance function is modified in order to change the weighting factor of each point. This factor is the inverse of the class imbalance in the train set. For example, in the binary K-NN between class label '3.0' and '4.0', the class label '3.0', which is 26.7% of the data, would get scaled up by 3.75 (1/0.266), while the class label '4.0', which makes up the other 73.3% of the data, would only be scaled up by 1.36 (1/0.733).

Then, from the three binary K-NN networks, the most common outcome for each data point is selected to be the final prediction. The use of multiple

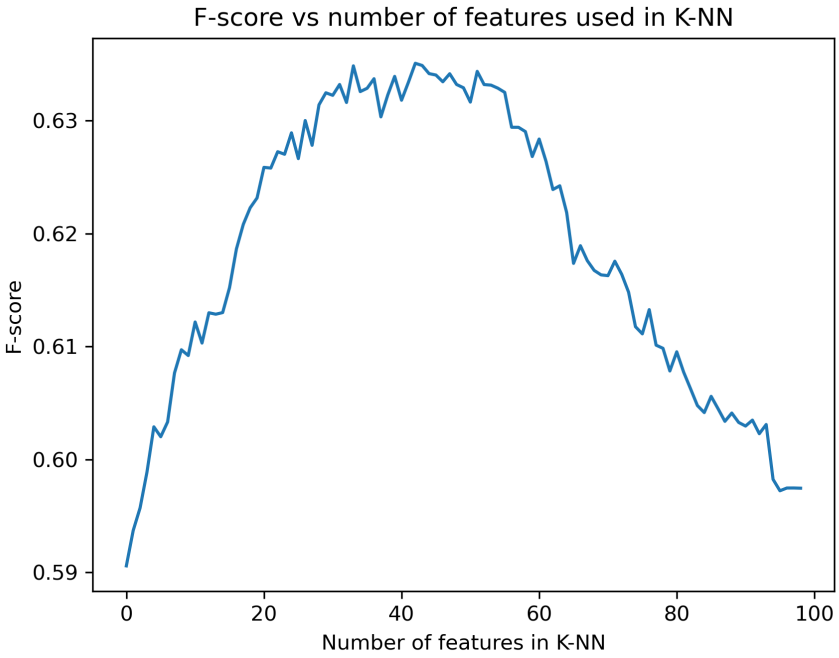

Figure 2

binary K-NNs is a form a stacking; which often can improve performance.

We can see in Table 2 the accuracy and F-score of the model, depending on which weighting algorithm is used. The use of inverse class label weighting causes the accuracy and F-score to fall, but the major class is at least predicted well. Ultimately, Model A is not the best model that can be produced.

4.2 Model B

Model B followed on from a process used by Lee, H., Park, M., and Kim, J. [3], who looked into using neural networks for Plankton classification, a highly imbalanced data sample. Amongst other ideas, random over-sampling (ROS) and random under-sampling (RUS) were utilised, along with 'Thresholding'. Thresholding is the concept of training the data on the full data-set, but only for a certain number of epochs (not necessarily letting the model converge). The reasoning behind this is to allow the model to understand the prior distribution of class labels, but not to a level where the class imbalance

Model	Accuracy	F-score	Major class
Baseline	0.690	0.626	0.912
ROS	0.584	0.597	0.611
RUS	0.467	0.508	0.445
ROS then Thres	0.667	0.639	0.819
RUS then Thres	0.673	0.638	0.838
Thres then ROS	0.602	0.609	0.650
Thres then RUS	0.560	0.583	0.571

Table 3: D-NN model comparison

causes issues.

Therefore 7 DNN are trained: A baseline, ROS/RUS only, Thresholding then ROS/RUS and ROS/RUS then Thresholding models. As we can see in Table 3, the base model over-predicts class label '4.0' due to class imbalance, and the ROS/RUS under-predict class label '4.0' as they do not know the underlying distribution of the class labels. The final 4 models, try to find the median between the two extremes of the first three models. We judge this on the F-score, which is a good measure between precision and recall. We can see the fourth model, which fits the model first on Randomly over-sampled data then thresholds it on the initial training data, has the highest F-score (0.639), so this is what is chosen as model B.

4.3 Model C

The lower level of model C is made up of three binary regression DNNs , with stacking of the three models using a classification neural network. We also utilise ROS/RUS and Thresholding techniques, which were found to work well in Model B. Due to time constraints, only the best 2 DNN from Model B were tested; which involved ROS and RUS then Thresholding. These models were fit onto the binary data, utilising the same structure as in model B, the only change being that the output was continuous, a value between 0 and 1 (the closer the number to 1, the more likely it is to be part of the minor class label). Along with the accuracy, since the models are binary, we can find the AUC to compare the models. AUC is an aggregate measure of performance that is often used in diagnostics of binary classification. In Table 4, we can see that RUS then Thresholding has a better accuracy for the majority of the binary models, and a better AUC for all of the binary models, so it was picked as the sub-model.

Model	Accuracy	AUC
'3.0' vs '4.0' ROS then Thres	0.736	0.603
'3.0' vs '4.0' RUS then Thres	0.736	0.617
'4.0' vs '5.0' ROS then Thres	0.938	0.570
'4.0' vs '5.0' RUS then Thres	0.936	0.616
'3.0' vs '5.0' ROS then Thres	0.870	0.718
'3.0' vs '5.0' RUS then Thres	0.871	0.744

Table 4: Binary NN sub-model comparison

Then, a stacked model was made: The input was the three continuous values between 0 and 1, the results of our binary models. For this, we used RUS/ROS then Thresholding too, in order to further combat class imbalance. As can be seen in Table 5, using ROS then Thresholding for the parent model gave us a greater F-score. This is thus chosen for model C.

Model	Accuracy	F-score	Major class
ROS then Thresholding	0.668	0.643	0.807
RUS then Thresholding	0.661	0.639	0.795

Table 5: Full Model C comparison

5 Conclusion

Three models were developed and evaluated, aiming to address the challenges that came with predicting the rating of books in the Goodreads dataset, namely class imbalance, and to a lesser degree, feature selection.

Model A employed a stacking approach of a modified K-nearest neighbour algorithm, in which the modification aimed to combat class imbalance. However, it performed the weakest of the three models, but did well at predicting high rating books. Model B utilised a deep neural network

with random oversampling and thresholding techniques. Model B had the highest accuracy of the three models, but still over-predicted the major class, although at a lower rate than a baseline DNN. Model C combined the strengths of Models A and B by using binary regression DNNs and stacking. The empirical results showed that Model C had a slightly higher F-score than Model B and is the best model produced in this project. Further investigations could look into larger DNNs where the training time is out of the scope of this project. Furthermore, different deep learning architectures such as convolutional neural networks could be explored and empirically tested to see if they enhance performance of the model.

References

[1] Dubey, H., and Pudi, V. Class based weighted k-nearest neighbor over imbalance dataset. In *Advances in Knowledge Discovery and Data Mining* (Berlin, Heidelberg, 2013), J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, Eds., Springer Berlin Heidelberg, pp. 305–316.

[2] Johnson, J. M., and Khoshgoftaar, T. M. Survey on deep learning with class imbalance. *Journal of Big Data 6*, 1 (Mar 2019), 27.

[3] Lee, H., Park, M., and Kim, J. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In *2016 IEEE International Conference on Image Processing (ICIP)* (2016), pp. 3713–3717.

[4] Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., Shyu, M.-L., Chen, S.-C., and Iyengar, S. S. A survey on deep learning: Algorithms, techniques, and applications. *ACM Comput. Surv. 51*, 5 (sep 2018).