Project report

Pratyay Piyush Mishra
DA25E053

Embedding generation

EmbeddingGemma-300 model was used to create embeddings for system prompt+user prompt , responses for each training data given.

Since Gemma embeddings are matryoshka embeddings, hence taking the 1st few dimensions along each vector is a valid transform and captures most of the important information encoded. We experimented with various embedding sizes and found that n = 64 gave us the best results for our model. Hence we truncated the input vectors to 64 from 768, thus reducing the risk of our models overfitting the data.

Data preparation

Apart from the 64*3 dimensional inputs for our prompt, response and metric embeddings, we also calculated the cosine similarity between prompt, response and metric embedding vectors(full sized) and the interaction terms between them(prompt_response cosine similarity*metric_response cosine similarity and so on..).

This gave us a total number of 64*3+7=199 features.

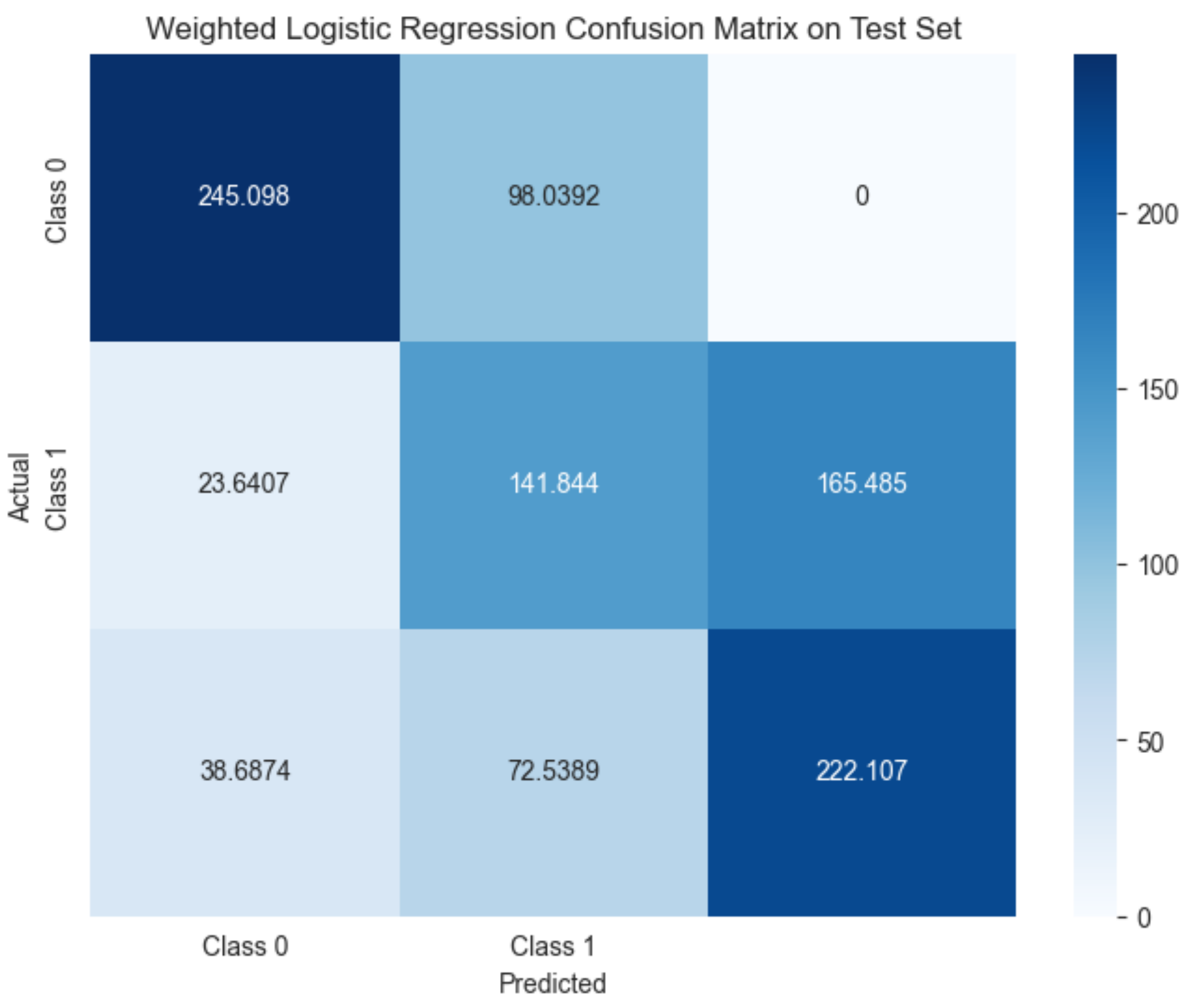
We also mapped the training scores to different bins as follows:

0-4:bin 0
5-7:bin 1
8-10:bin 2

Logistic classification

Since the number of data points having a low score was very low, we used weighted logistic regression, where class weights were assigned to each sample based on number of instances of its class. This made minority classes have higher weights and thus avoided the classifier predicting majority class for all samples.

The confusion matrix for the trained model on the test split is given below:

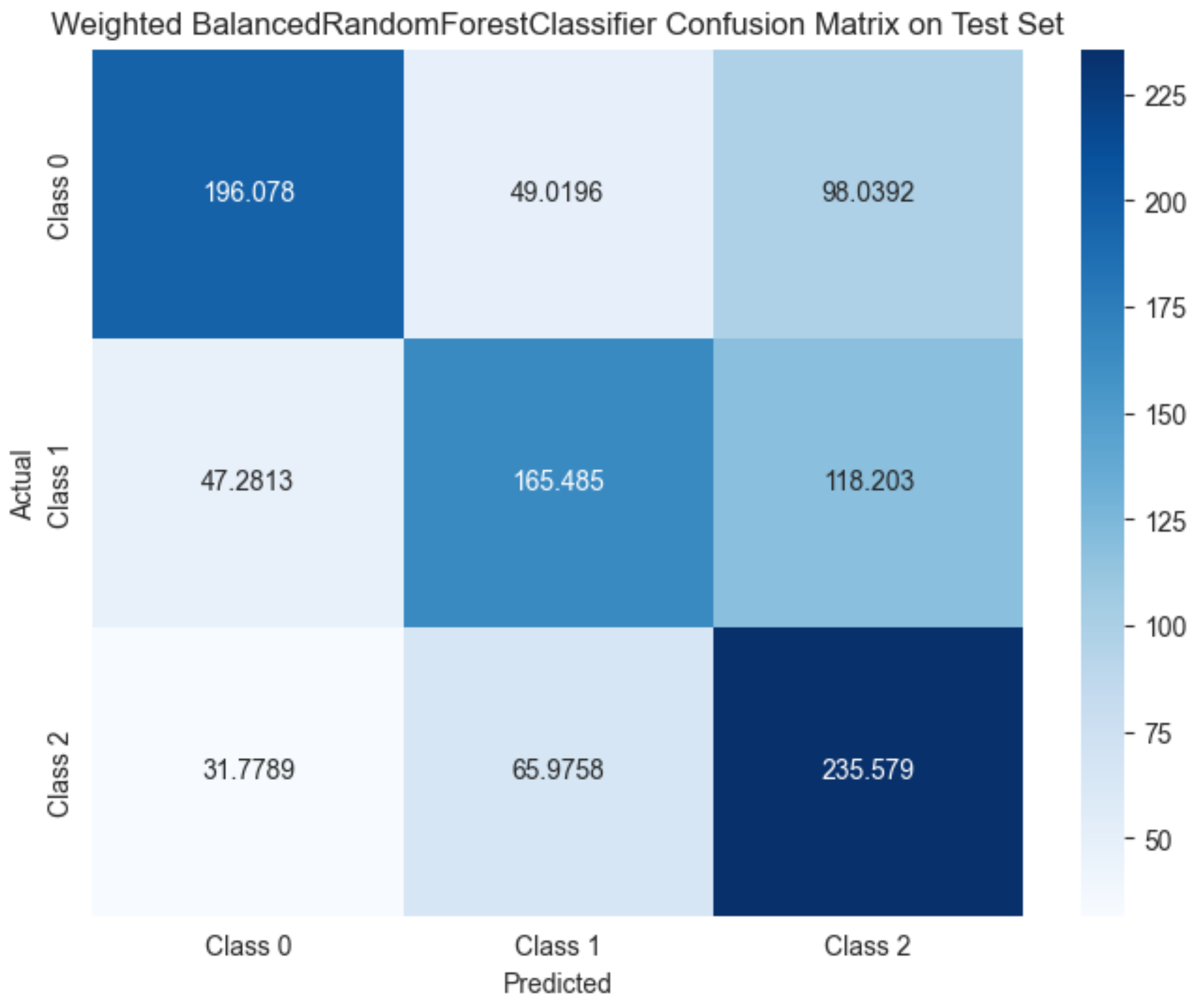


As we can see, the model was able to differentiate between class 0 and 2 well, signaling that our sample can distinguish between bad and good samples well.

BalancedRandomForestClassifier

We then tried using a tree based model, since the dataset was unbalanced, we used a balanced random forest model, that takes class weights as input, and takes a bootstrap sample from the minority class and samples with replacement from the majority class while sampling.

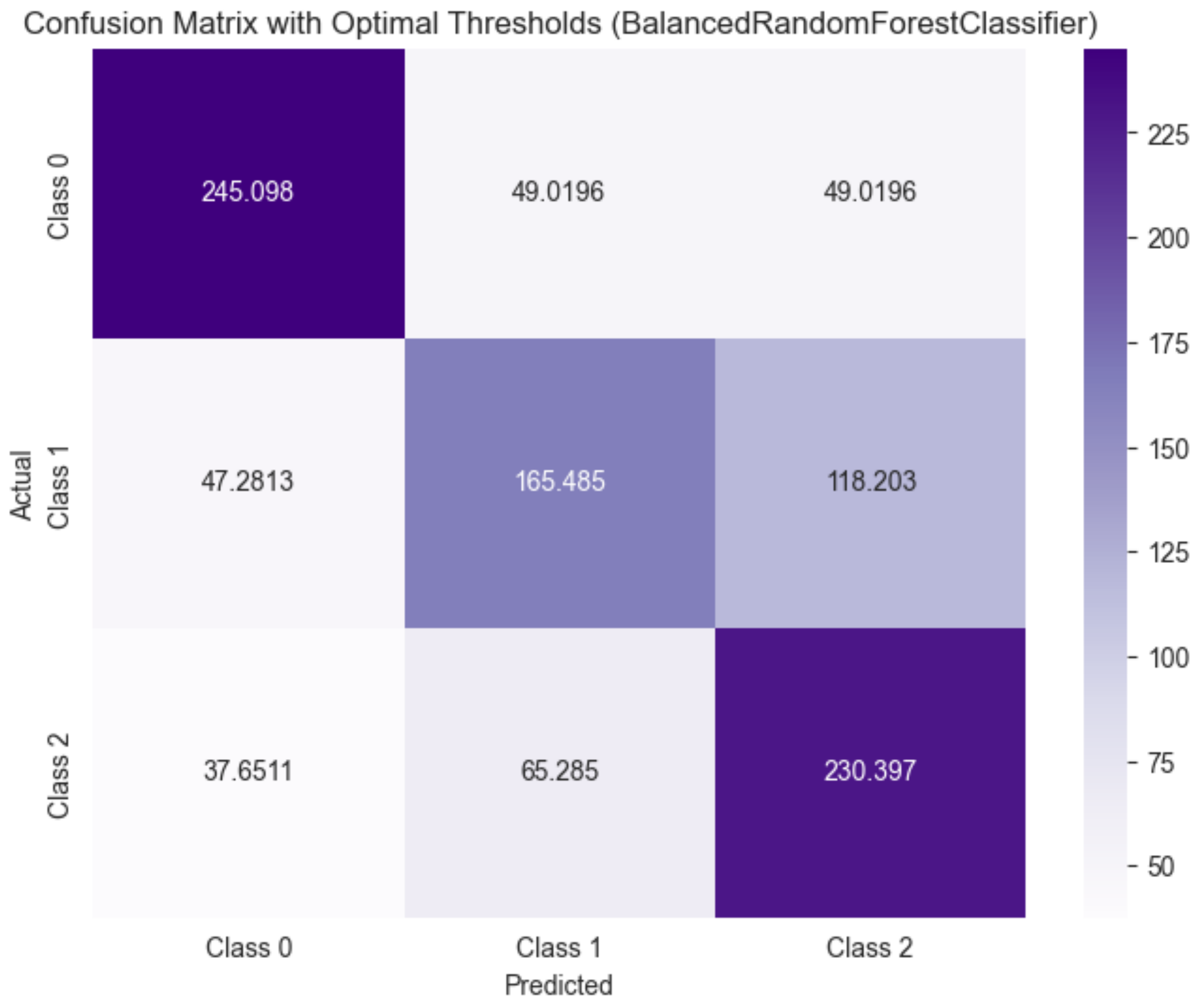
This then was used to train a new model, which gave below confusion matrix:



We then tuned for classification probability thresholds for minority classes based on f1-score, and selected below thresholds for class 0 and 1:

Optimal Thresholds found based on Weighted F1-score:
  Threshold for Class 0 (t0): 0.35
  Threshold for Class 1 (t1): 0.50

The updated confusion matrix based on these thresholds was as below:



We can see that the performance on class 1 improved slightly, but the classifier is still unable to distinguish between moderately good and great prompt response pairs.
We then scale the output back to scores as per below:
0 mapped to 4
1 mapped to 6
2 mapped to 7

We keep the scores for classes 1 and 2 close because this classifier is still unable to differentiate well between these 2 classes.
This model performed the best with a score of 3.605 on the private test set.

Final model performance statistics

Classification Report with Optimal Thresholds:

```
         precision   recall  f1-score
    0       0.74      0.71     0.73
    1       0.59      0.50     0.54
    2       0.58      0.69     0.63

accuracy                        0.64
```