4.1)

The GoEmotions dataset given on Moodle contains around 58,000 Reddit comments labeled with one of the 28 emotions and one of the 4 sentiments. After plotting the distribution of the comments' labels, we can see in the Figure 1 below that there are fewer "ambiguous" comments than in the other sentiment categories. The rest of the comments are pretty well distributed in the three other sentiments.

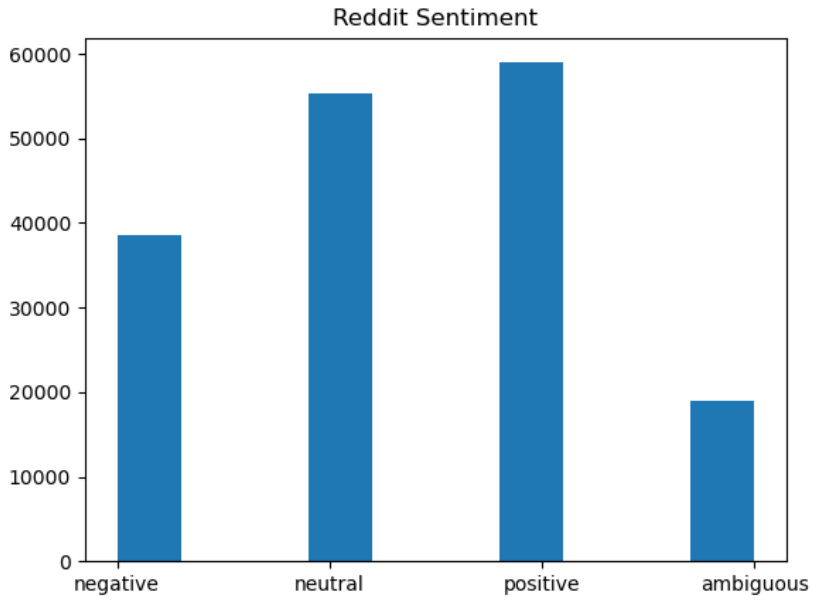

Figure 1: Sentiment distribution

This imbalance could impact the training process in a way that the classifier considers the "ambiguous" category to be less common. This bias could then impact the prediction process in a way that comments will have a smaller probability of being labeled as "ambiguous" when they should be. In the Figure 2 below, we can see a huge imbalance in the emotions category.

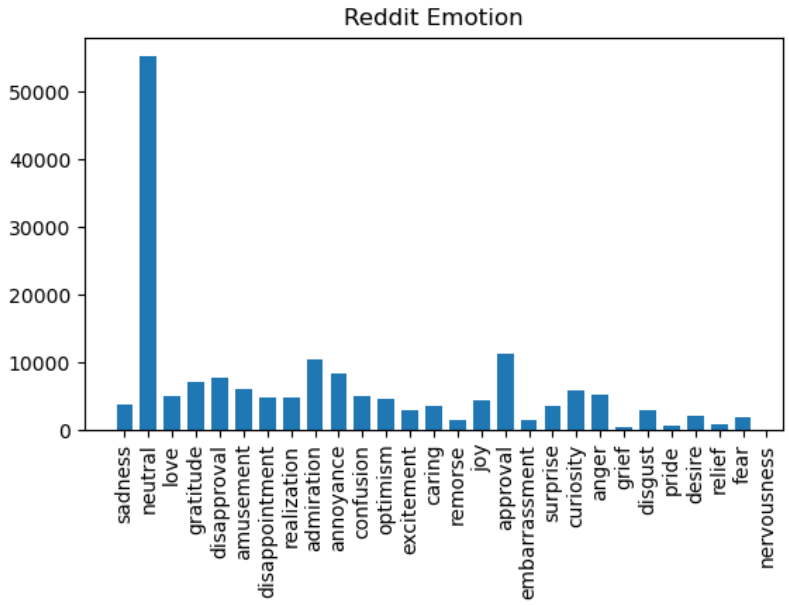

Figure 2: Emotion distribution

There are excessively more comments labeled as "neutral" than in any other category. Since the classifier has been trained with way more neutral comments than with any other category, it could bias the prediction process to wrongfully label more comments as "neutral". Due to this imbalance, not any metric can be used to measure the emotion classifier's performance. Indeed, it will probably seem very performing thanks to its high emotion accuracy as it predicts correctly for the majority class ("neutral"), but the F1-score, which balances prediction precision and recall, will be way lower. Hence, F1-score might be a better performance indicator for emotion prediction. In the case of sentiment prediction, F1-score will probably still be lower than the accuracy, but way closer. Hence, the choice of metric in that case might be less important than with emotion.

4.2)

Words as Features:

The first part of the mini-project was to train various classifiers with words as features vectorized with one-hot encoding. Here's a table of the various emotion classifiers' performance:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Base-MNB	Default	0.389	0.167	0.311	Some classes don't have predicted samples in the

					classification report.
Base-DT	Default	0.360	0.278	0.361	
Base-MLP	Default	0.373	0.288	0.373	Hasn't converged after 200 iterations.
Top-MNB	Alpha = 0.5	0.393	0.223	0.348	Some classes don't have predicted samples in the classification report.
Top-DT	Criterion = gini Max depth = 4 Min samples split = 2	0.375	0.082	0.229	Some classes don't have predicted samples in the classification report.
Top-MLP	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.435	0.275	0.388	Hasn't converged after 3 iterations.

Table 1: Emotion classifier performance (words as features)

We can see that the Top-MLP classifier, which uses the ReLU activation function, the Adam solver, 3 maximum iterations, and two layers of 30 and 50 neurons, is the best one as it has the highest Accuracy, which is 43.5%, and highest Weighted-average F1, which is 38.8% [Table 1]. In the case of this highly imbalanced emotion dataset, Weighted-average F1-Score is the most important metric to consider. Indeed, the fact that it is weighted according to the individual sample sizes makes it more accurate to compare the actual performance with other models. Also, since the F1-score is a mix of precision and recall measurements, it is a better overall indicator of performance. Another important aspect to note is that the Top-MLP and Base-MLP classifiers did not converge after their specified maximum number of iterations. In order to make the model training more time-manageable and to be able to train all the classifiers before the assignment deadline, I had to limit the Top-MLP maximum iterations number to 3. By default, Base-MLP runs until it has reached 200 iterations. Base-MLP not converging after 200 epochs means that it might have taken way more iterations to return a converging multi-layer perceptron. In that scenario, the accuracy and F1 scores might have been even higher. Nonetheless, we can consider our best Weighted-average F1 score of 0.388 to be very low. This might have been caused by many reasons. First, our model might have needed many more epochs in order to converge. Second, the one-hot

encoding method to vectorize our dataset might not be the best method to extract word features. Third, our dataset might not be large enough to provide enough instances of each emotion class. This third reason can be assumed thanks to the fact that many of the classifiers returned a "labels with no predicted samples" error when retrieving the classification report. This means that the test set did not have any instances of certain classes to classify. By analyzing the performance file, we can see that "nervousness", "grief", "relief", and "pride" were never tested in the performance report. This is a very important indicator that the high quantity of classes lowers the number of instances per class, except for the "neutral" class which is extremely imbalanced. A fourth reason why our performance might be so low is the choice of possible parameters and values.

Here's a table of the various sentiment classifiers' performance:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Base-MNB	Default	0.297	0.251	0.290	
Base-DT	Default	0.274	0.250	0.278	
Base-MLP	Default	0.268	0.252	0.286	Hasn't converged after 200 iterations.
Top-MNB	Alpha = 0.5	0.542	0.503	0.537	
Top-DT	Criterion = gini Max depth = 4 Min samples split = 2	0.384	0.213	0.280	
Top-MLP	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.572	0.538	0.569	Hasn't converged after 3 iterations.

Table 2: Sentiment classifier performance (words as features)

It can be seen that the Top-MLP classifier is again the best way to classify words as features. Indeed, it provides the highest accuracy (57.2%) and F1-scores (53.8% Macro-average F1 and 56.9% Weighted-average F1) with its ReLU activation function, Adam solver, 3 maximum iterations, and two layers of 30 and 50 neurons [Table 2]. The sentiment labels are not as imbalanced as the emotion dataset, so the use of the Accuracy metric is indicative enough of the model's performance. Another important aspect to note is that our best-performing model, the Top-MLP, has again not converged with the maximum number of iterations provided. The model might have needed a higher epoch limit to yield better performance. In that case, the Accuracy and F1 scores might have been even higher. The Accuracy of 57.2% is way better than the emotion's multi-layer perceptron, but still a bit low [Table 2]. This might have been caused by the low number of maximum iterations limit, the small dataset size, or the wrong training and testing set

split ratios. At least, the sentiment dataset is not as imbalanced as the emotion one. This means that the classifier has been trained on somewhat equally distributed classes and that the classification report was able to test a sufficient number of instances of each class. However, as displayed in the performance file and as expected due to its slight imbalance, the "ambiguous" class has the lowest Precision (44.8%), Recall (34.5%), and F1 (38.99%) scores [performance.txt]. This means that it was less accurate in correctly labeling "ambiguous" instances when it had to.

When experimenting with the same types of classifiers but trained with words as features and stop words removed, I was surprised to find out that the classifiers' performances were the same. Initially, I would have thought that removing stop words would have removed unessential noise, hence yielding better training results, but it seems not. One of the possible reasons is that stop words are not too present in the dataset, hence not producing too many unnecessary features. Another cause might be that the model is able to identify and understand that the stop word features have no real impact on the output label, and therefore automatically reduces the weight given to them during the training process.

Embeddings as Features

The second part of the mini-project was to train various types of multi-layer perceptron classifiers with word embeddings as features.

Here's a table of the various emotion classifiers' performance:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Base-MLP	Default	0.402	0.169	0.304	Hasn't converged after 3 iterations. Also, some classes don't have predicted samples in the classification report.
Top-MLP	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.398	0.152	0.293	Hasn't converged after 3 iterations. Also, some classes don't

					have predicted samples in the classification report.

Table 3: Emotion classifier performance (words embeddings)

Unexpectedly, the Base-MLP classifier with the maximum number of iterations set to 3 and the rest of the parameters set to the default values has performed slightly better than the other custom multi-layer perceptron models. However, various factors could have yielded this result and it can be concluded that both models performed equally. With an Accuracy of 40.2% and a Weighted-average F1 score of 30.4%, it still has not beaten the performance of the Top-MLP classifier trained on word as features vectorized with one-hot encoding (43.5% Accuracy and 38.8% Weighted-average F1 score) [Table 1 and Table 3]. Again, the Base-MLP classifier never converged, so having a much higher maximum number of epochs might have yielded even better results. The classification report also returned the same error where some classes never had predicted samples. This is a confirmation that the dataset imbalance is highly affecting the training process.

Here's a table of the various sentiment classifiers' performance:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Base-MLP	Default	0.531	0.469	0.519	Hasn't converged after 3 iterations.
Top-MLP	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.531	0.466	0.519	Hasn't converged after 3 iterations.

Table 4: Sentiment classifier performance (words embeddings)

It can be seen here that the Base-MLP and Top-MLP classifiers are performing extremely similar in terms of Accuracy and F1 scores. It could have slightly varied depending on the training samples inputted and the training and testing split ratio, but it would still be very similar. After analyzing the results of the various MLP classifiers trained on embeddings with emotion and sentiment classes, we can conclude that the choice of parameters might not have huge importance when working with embeddings on multi-layer perceptrons. Indeed, the reason could either be that all the models have reached their equal peak performance on that specific dataset and that other types of classifiers should be used, that the embeddings on that specific dataset do not provide enough diversity of vectors to yield a meaningfully better performing combination of parameters, or that the pre-trained "word2vec-google-news-300" is not performing well enough. However,

none of the classifiers converged, so the results might have been very different if given way more time.

Here's a table of the various emotion classifiers' performance with other pre-trained embedding models:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Best MLP (glove-wiki-gigaword-200)	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.365	0.1	0.25	Hasn't converged after 3 iterations. Also, some classes don't have predicted samples in the classification report.
Best MLP (glove-twitter-50)	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.352	0.08	0.23	Hasn't converged after 3 iterations. Also, some classes don't have predicted samples in the classification report.

Table 5: Emotion classifier performance (embeddings with other pre-trained models)

Here's a table of the various sentiment classifiers' performance with other pre-trained embedding models:

Name	Parameters	Accuracy	Macro-average F1	Weighted-average F1	Notes
Best MLP (glove-wiki-gigaword-200)	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.484	0.02	0.51	Hasn't converged after 3 iterations. Also, some classes don't

					have predicted samples in the classification report.
Best MLP (glove-twitter-50)	Activation = relu Hidden layer sizes = [30, 50] Max iter = 3 Solver = Adam	0.470	0.02	0.5	Hasn't converged after 3 iterations. Also, some classes don't have predicted samples in the classification report.

Table 6: Sentiment classifier performance (embeddings with other pre-trained models)

After running various experiments on the same classifiers but using different pre-trained embeddings models, we can see that the choice of pre-trained embeddings model has a real impact on the MLP classification performance. Indeed, for the emotion dataset, the "word2vec-google-news-300" yields a Weighted-average F1 score double that of the "glove-wiki-gigaword-200" and "glove-twitter-50". This probably means that the greater the dimension of the embeddings vector, the better the result. Another reason for that high-performance difference might be the quality of the pre-trained embeddings model. Indeed, the word2vec-google-news-300 model has been trained on 3,000,000 records while the glove-twitter-50 and glove-wiki-gigaword-200 models were trained on 1,193,000 and 400,000 records respectively. This is a clear indicator that a larger training set on an embeddings model directly produces a better-performing multi-layer perceptron.

4.3)

As the sole member of my team, I was responsible for completing all the tasks in the assignment. Therefore, I contributed to all the work myself.