Abstract:

In this assignment, we evaluated the accuracy of the logistic regression model on a dataset of positive and negative IMDB reviews. We selected the 300 features out of a pool of 89526 by eliminating all non-important words, and especially by using the z-score. We found that using a low learning rate gave us very good results, especially concerning the convergence of the loss function, and that the logistic regression model gives a higher AUC (0.92) when compared to the KNN model. These results were found to be consistent even when changing the training to testing dataset size ratio.

We also investigated the use of a multiclass regression model to classify texts/documents based on the words that appear in them. We selected words based on their mutual information with the corresponding text's class and used these words to train the model. We investigated this model's performance in terms of accuracy and number of iterations required to converge (thus computation). We determined that the accuracy of this model is greater than that of a KNN model with the same training data (up to a maximum of 40 neighbors considered in calculations), regardless of how much training data was used. We also determined that the number of iterations required to converge varies greatly based on the learning rate we used. This learning rate could vary from 1% to 40%, and the models converge to similar results on cross entropy in training/validation datasets. With a higher learning rate, the number of iterations (thus computation) was greatly reduced, but the cross-entropy graphs generated look unnatural and could lead to problems such as the model never converging or converging on sub-optimal values.

Introduction:

We were tasked in this project to explore linear classification and to examine the selection of features based on their linear coefficients. The first dataset used in the experiment is that of the IMDB reviews, which contains the rating of 50000 movies along with the words used in these reviews. The most important words were chosen based on their z-score, and the words with the 150 lowest and 150 highest z-scores were selected as features. The goal was to predict in which category would a movie fall into based on its features ($y = 1$ would classify as a good movie, and $y = 0$ would classify as a bad movie). We did so using logistic regression, and we got an accuracy was of 0.825 on the testing data as compared to 0.829 on the training data. Furthermore, we compared our results with the KNN model, and we consistently found that our model gave a better accuracy, especially when examining the results of the area under the curve for both models. Finally, we also conducted an experiment with Lasso regression to see if the features chosen with the Lasso method would differ from what we have gotten using the z-scores.

We also performed multiclass regression on sklearn's 20 news groups dataset. This dataset consists of multiple texts/paragraphs each belonging to a class. We decided to select 4 of these 20 classes, which are 'comp.graphics', 'rec.sport.hockey', 'sci.med', and 'soc.religion.christian'. We then used multiclass regression to predict which class a text belongs to. The multiclass regression was performed using the frequency of the words present in that text as features. Since a text may contain many words, we used mutual information to select words which are more present in a text of that class. To do so, we calculated the mutual information between each word in our training samples and that sample's classification. Mutual information is a measure of how similar two labels from the same data are (2). However, when training the model itself, we used term frequencies times inverse document frequency (Tfidf) rather than occurrences. Tfidf is a measure of a term's frequency in a document multiplied by the inverse of that term's frequency in all provided documents. One reason for doing so is that a longer text is expected to have more occurrences of any words than a shorter text, although this has no impact on what class the text belongs to (1).

Datasets

The IMDB dataset consists of 50000 reviews separated into testing and training datasets, each containing half of the files. The datasets themselves are also divided based on the rating each movie has gotten. Thus, a movie with a rating of 7 to 10 (inclusively) would be classified as a positive review, and a movie with a rating of 0 to 4 (inclusively) would be classified as a negative review. We were also provided with a file containing the entire vocabulary used in the reviews, along with

the frequency of each word in each review. Since the total vocabulary size was of 89526, we needed to remove rare and stop words, which would appear respectively in less than 1% or in over 50% of the documents. We were left with 1744 features, of which we chose 150 features with the most positive z-scores, and 150 features with the most negative z-scores.

Overall, the 4 classes we selected from the 20 news groups dataset consisted of 2377 samples. Each of these samples is a paragraph in English, and as is the case with any paragraph, there are a lot of basic words that are used in most contexts and don't help predict the class of a text. To filter those out, we filtered words which appeared in less than 1% of all documents and words that appear in over 50% of all documents. We then noted the number of occurrences of each remaining word in each document and used this information to calculate the mutual information score between each word of a text and that text's class. We used this information to select the top 100 most relevant words for predicting each class. Since some words are used for multiple classes, we ended up with 297 distinct words. We used these words as the features we were going to look for. When training and using the model, we used term frequencies times inverse document frequency (Tfidf) rather than occurrences in the document for the selected words. A benefit from doing so is that when training the model with occurrences, some terms with high occurrence in a text gave rise to large numbers when multiplied with features weights (w), which, when exponentiated, result in overflow during prediction. It has been pointed out that we can simply subtract the maximum value in that matrix from all elements and the softmax function still works as intended, but when doing so, we ended up with very negative numbers which, when exponentiated, approximate to 0. Not only are these non-informative, but when taking the logarithm for cross-entropy, we get that log(0) = NaN which breaks the code.

Results

IMDB:

When calculating the z-score for every single feature for the IMDB experiment, we were able to find the features with the 10 most negative z-scores, and those with the 10 most positive z-scores based on the training dataset as can be seen in figure 1.

However, for the rest of the experiment, we used 300 features in total, which gave us consistently a higher accuracy compared to the KNN model based on the AUC and the AUROC:

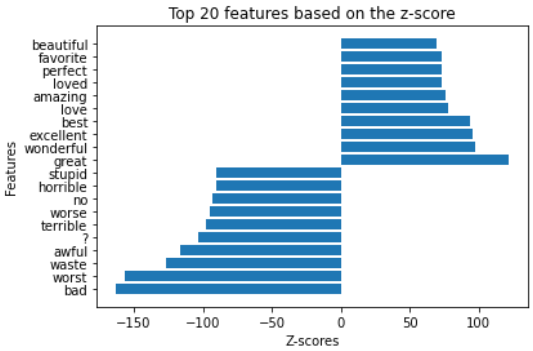

***Figure 1**. Top 20 features based on the z-score*

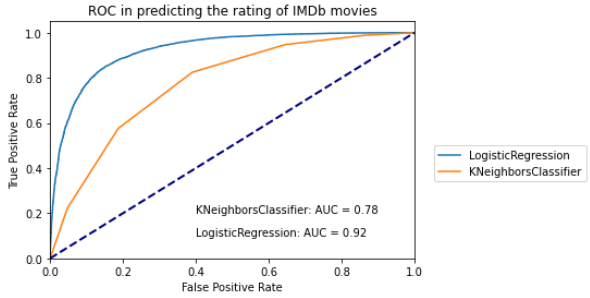

***Figure 2.** ROC (Receiver Operating Characteristic) curve in prediction the rating of IMDB movies. The test set represents 80% of the data used prediction.*

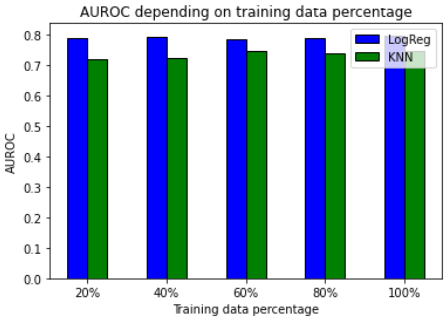

***Figure 3.** AUROC depending on training data percentage*

Figure 3 represents the ROC in predicting the rating of IMDB movies (either y = 1 for a good movie or y = 0 for a bad one) when using 20% training data, and we can see that the area under the curve for logistic regression is higher than that of KNN (0.92 compared to 0.78 for KNN). We can see this trend even when varying the size of the training data as shown in the bar plot. While the accuracy of KNN seems to increase the more we increase the size of the training dataset, the accuracy of the logistic regression model hovers around the same value, which could demonstrate its consistency as a model.

Also, we chose the learning rate based on the results we got from plotting the cross-entropy as a function of iteration. We can see clearly that the function converges in a very small number of iterations in the first graph, which shows that the model was correctly implemented and that the learning rate of 0.00005 is appropriate. It is also important to note that the monitoring the loss was a lot slower when using a higher learning rate (0.05), and that it necessitated a significantly higher number of iterations. We show in the figure 5 the result of using a learning rate of 0.05 with a number of iterations of 1000.

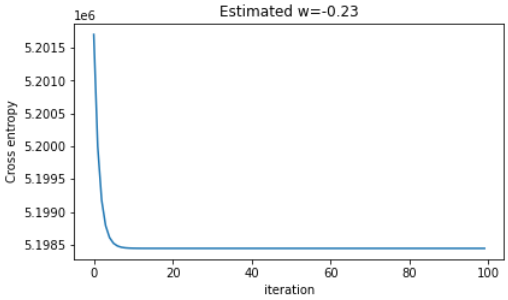

Figure 4. *Cross-entropy as a function of iteration with a learning rate of 0.00005.*

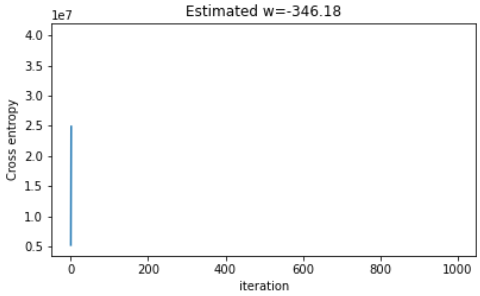

Figure 5. *Cross-entropy as a function of iteration with a learning rate of 0.05*

We also checked the gradient using small perturbation, and we found a difference of 6.621049148397399e-18 between the approximated gradient and the hand calculated one.

Moreover, we plotted a bar graph representing the weight of each of the top 20 features, and we can see in figure 6 that the negative features have a larger weight in terms of absolute value. Indeed, this could indicate that these features are the most substantial in choosing the appropriate label for the testing data.

Finally, we found using the sklearn implementation of Lasso regression that only 3 features were the most important: 'bad', 'worst', 'great'. This aligns with the results found using our model as these 3 features have the highest z-scores in terms of absolute value as we can see in figure 1.

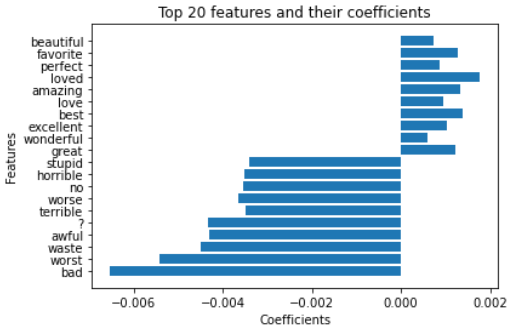

Figure 6. *Top 20 features and their coefficients*

Multiclass:

This graph shows the cross entropy of the model decreasing as a function of iterations. We can see that the cross entropy for the training dataset tends to decrease more rapidly than that of the validation dataset. In our model implementation, we decided to stop the training of the model when we see a cross entropy value in the validation set which is larger than the mean of the past 10 calculated cross entropy values to avoid overfitting. In the above plot, we see it happen at iteration 1640 when using a learning rate of 0.01 or 1%. At this point we know the model's w values have converged on the optimal weights (based on our data). We also tried using learning rates of 2% or 3%, both of which worked and resulted in the model converging after around 800 and 500 iterations respectively. The graphs for these look similar to the graph above with the only difference that they stop earlier.

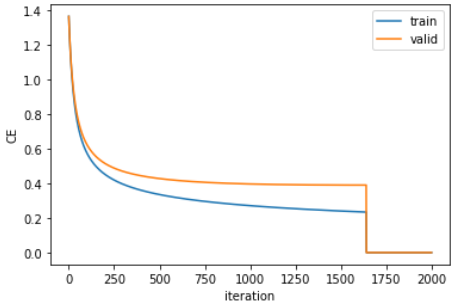

Figure 7. *Cross entropy of the model decreasing as a function of iterations. The learning rate used is of 0.01.*

The trend observed holds, but with higher learning rates, the graph looks unnatural and converges much quicker. For example, with a learning rate of 0.4 (40%), the graph converges after 37 iterations as we can see in figure 8. We can see that the cross entropies still converge to approximately the same values, meaning this fitted model is similar to the previous one. However, this larger learning rate led to much faster fitting due to the drastically lower number of iterations (thus computation) required for converging. We also see that the cross entropy first increased before decreasing.

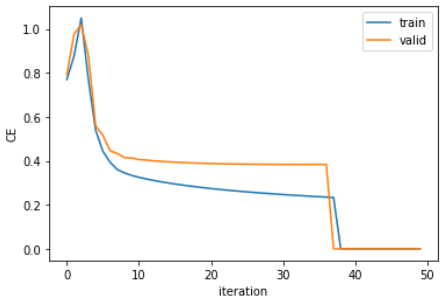

Figure 8. *Cross entropy of the model decreasing as a function of iterations. The learning rate used is of 0.4.*

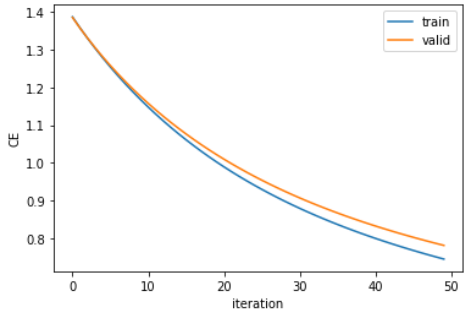

Figure 9. *First 50 iterations of the cross-entropy curve with a 1% learning rate (first 50 iterations of the curve represented in figure 7).*

We can compare the first 50 iterations in figure 8 with the first 50 iterations of the cross-entropy curve with 1% learning rate in figure 9 and see that the slower learning rate leads to a much smoother, although much less steep, curve.

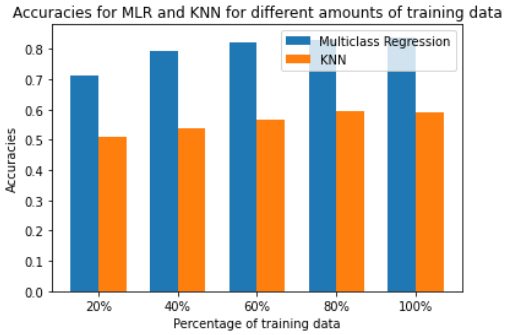

Figure 10. *Bar plot representing the accuracies given by the KNN and multiclass regression based on different portions of the available training data*

The bar plot shows the different accuracies of KNN and multiclass regression based on different portions of the available training data. In all cases, multiclass regression provides more accurate predictions of data labels when compared to KNN. KNN was fitted with the highest accuracy obtained when trained with the training data and verified with the validation set using from 1 to 40 neighbors. This most accurate number of neighbors (up to 40) was used to fit the model and generate the results in the above plot. Furthermore, as more data is provided, we see greater improvements in multiclass regression classification accuracies when compared with KNN classification accuracies increasing.

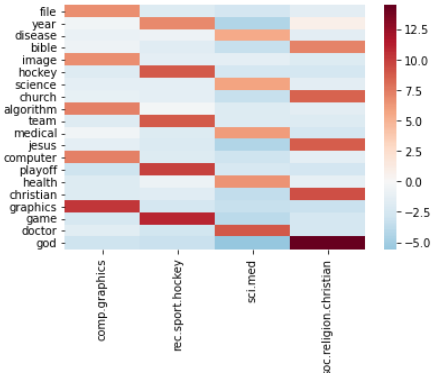

Figure 11. *Heatmap showing the different values of w for the top 5 words of each class.*

The heatmap shows the different values of w (linear coefficients) for the top 5 words of each class. The words are on the y-axis and the classes are on the x-axis. We can see that there is no overlap between these words for different classes, each of these words is specific in predicting a single class.

Discussion:

We can see from the experiments done on the IMDB datasets that logistic regression is consistently more effective than KNN as we get a higher accuracy even when modifying the training to testing set ratio. It would be interesting to see if these results are consistent when doing unsupervised or semi-supervised machine learning. We saw that the logistic regression model we have implemented can support a very low learning rate, but that a much higher one requires significantly more iterations and does not result in the convergence of the loss function. This is indicative of the gradient of the loss function being a lot steeper, and thus, we cannot afford to produce a bigger step size as it would make our function diverge. Moreover, it was interesting to see the weights our top features have gotten, and how much higher they were for the negative coefficients. It would make sense for the model to use these words to predict accurately in which category the movie would fall into, especially since words such as "worst", "bad" and "waste" are strongly indicative of a movie having a very low rating. In contract, some of the more positive features did not have as high of a coefficient, and perhaps this is because when a movie is being reviewed positively, more diverse words would be used to describe different aspects of that film. It would thus be very interesting to see if the choice of vocabulary is indeed more diverse when a movie is being given a more positive rating in future experiments. Finally, we conducted an experiment with Lasso regression as to see whether the features chosen by the model would differ from our results. We found that only 3 features were the most important according to Lasso regression: "bad", "worst" and "great". These results make sense since saying that a movie is "great" would automatically mean that it should have a high rating and therefore be in the "good" movie category, while the adjectives "bad" and "worst" are clearly pejorative and are indicative of a bad movie. However, in terms of predicting whether the movies in our database are good or bad, it would not be effective to only use these three features since it is not guaranteed that they will appear in the reviews, and especially because other words could be used to illustrate the quality of a movie. Hence, while the Lasso regression outputs interesting results that corroborate our findings, we believe that using the logistic regression is still a better choice for this experiment as it allows for more flexibility in terms of the feature selection.

Furthermore, the above experiments demonstrate that multiclass regression is more effective in predicting classification based on textual data on the 20-news group dataset when compared to KNN. It would be relevant to further test this finding on different textual datasets to see if the results we observe are consistent throughout different data samples (ie if multiclass regression is usually more accurate than KNN for classification of texts). We also determined that our multiclass regression model is capable of training with very large learning rates (up to 40%) with the training data we provided and still converge on very similar values as those obtained with a 1% learning rate. This provides the capacity for training the model in a small number of iterations (we observed 37) which can be done very quickly. Once again, it would be relevant to determine if other multiclass regression models can support higher learning rates, and how high these learning rates go. It would also be beneficial to determine an "optimal" learning rate which reliably fits the model to optimal weights in as little iterations as possible. If fitting the model is to be done multiple times in some setting (such as performing cross-validation), then obtaining lower fitting iterations can save computation and time.

Contributions:

Soumaia did all the logistic regression/IMDB dataset and Arien did all of the multiclass regression/20 news groups dataset.

Citations:

1. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer
2. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mutual_info_score.html