Classification of Textual Data

Group 5: Adam Motaouakkil (260956145), Frédéric Mheir (260636214), Yann Bonzom (260969653)

Abstract

This project focused on analyzing the data of IMDB reviews using logistic regression and the 20NewsGroups data using multiclass logistic regression. With logistic regression, the task was to classify articles as positive or negative reviews based on the words used; for multiclass regression, the task was to classify articles in different categories based on their contents.

Logistic regression was performing around AUROC values of 0.84 to 0.9. The higher accuracy values were performed when modulating learning rates, epsilon for gradient norm, and maximum iterations. Using a separate function to optimize these parameters more completely would have significantly increased runtime which is unfeasible on our resource scale, so individual values were used to test the model. The 20 most significant words by coefficients also differed from the linear regression choices, but many of them were similar. However, the choices could easily change depending on gradient values.

Multiclass regression performs at around 73%. Increasing maximum iteration and learning rate improved the model's performance. It was efficient at categorizing articles as words that indicated a certain article were recognized and influenced the model's predictions. Some sections were better predicted than others. KNN might be a better choice than multiclass, as it yields better accuracy; logistic, on the other hand, is a better alternative to KNN for binary classification given its hyper-parameterization potential.

Introduction

The IMDB dataset came in a folder containing articles as well as a *.vocab* file that indexes the *.feat* file for the words used in the articles. The 20NewsGroups article was pulled via a Scikit-Learn function. The preprocessing task was to clean up the words to remove irrelevant features. For logistic regression, the main findings were that the choices for stopping rate were strongly influencing accuracy values for predictions and that it performed well in predicting whether a review was negative or positive. *Li et al. (2015)* describes how 20NewsGroups can be filtered appropriately through a set of library functions and a description of the dataset, so we replicated this.

The main objective of the assignment is to investigate how pre-processing affects classification and helps model non-linear data. As mentioned by *Maas et al. (2011)*, unsupervised ML models are not very efficient in recognizing the strength of emotion in a word. Supervised models therefore definitely perform better, but this remains a difficult task. Afterall, it is the model creators who decide if a certain set of words is appropriate for use in modeling.

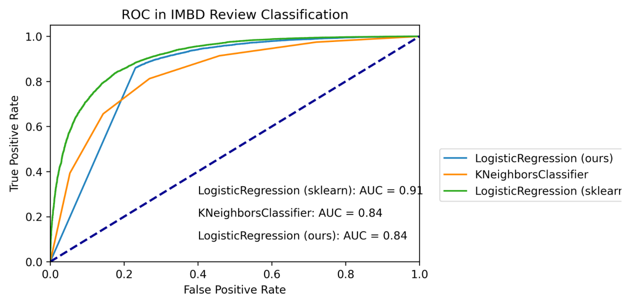
Datasets

The IMDB dataset consists of 25,000 training and 25,000 testing data points, each datum being a film review and its rating out of 10. We performed the following processing steps: (1) removed stop words (words appearing in >50% of the documents) and rare words (< 1%) from the training set; (2) converted the ratings to binary (0 = negative, 1 = positive) for both training and testing sets; (3) calculated the testing set standardized z-score correlations via simple linear regression between each word and the ratings to then determine the top 100 words that provide the strongest indication of the output; (4) create the final testing set by choosing only those same top 100 words columns and standardizing them. This z-score approach to determine the most important features indicated negative and positive reviews (e.g., words such as “bad” and “perfect”).

For the 20NewsGroups dataset, the processing differed as we set it up for multiclass predictions. To lighten runtime load, we decided on 'comp.graphics', 'rec.sport.hockey', 'sci.med', as well as 'soc.religion.christian', resulting in 2377 documents in total. To handle the data, we used a CountVectorizer() to vectorize the data and then convert this into a Pandas dataframe to store all the word occurrences across all documents. Using this, we removed words occurring in >50% and <1% of the documents (given that we have 4 categories, even words that appear exclusively in all documents of 1 class would occur in 25% of the data; this way, words that are potentially very indicative of a certain class are still preserved). Next, we one-hot-encoded the categories to get a 2377 x 4 matrix. To determine the most important words for each category, we calculated the mutual information scores between each word and each class; the resultant collection of words seems highly indicative of the chosen categories (e.g., “graphics” for comp.graphics, or “christianity” for soc.religion.christian) and we chose 15 words per category for faster training. Taking the union of word sets gave us 59 words in total. We standardized the inputs, and created the testing dataset by count vectorizing, choosing the same 59 words as our features, and standardizing it.

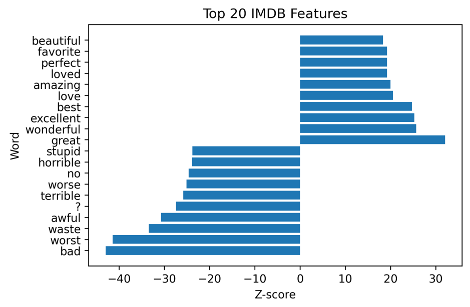
Results

Logistic Model Results

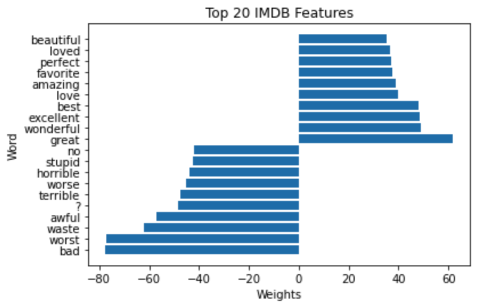


To support our model, we computed the AUROC for the logistic regression (above). We can see that it performs similarly to the SKlearn version, which indicates that it is well implemented.

Top Words (Z-score & Weights)

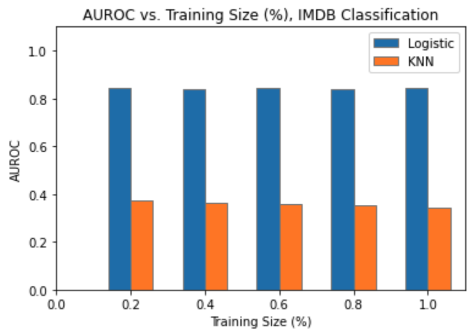


The z-scores of the top 20 words (10 most negative, and 10 most positive z-scores) were found by fitting every word column across all documents to the output column (indicating whether a review is positive or negative) via linear regression. This was done following the formula of *z_score* = $\mathbf{x}^T\mathbf{y}$ / sqrt(N). The results are as expected: words with negative connotations such as “bad” and “worst” work as indicators for bad reviews, and the reverse goes for positive words such as “beautiful” and “favorite”. However, the word “?” stands out. We decided to keep the question mark as it works as an indicator of confusion, hence its highly negative z-score.

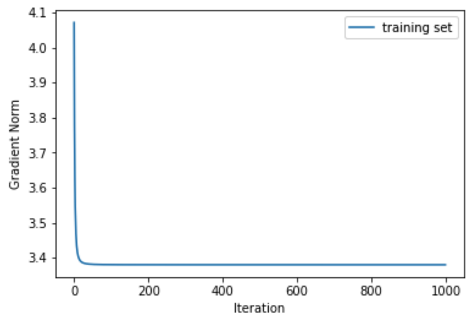


The weights of the top 20 words were found after the logistic model converged on the IMDB dataset. Throughout every iteration of the gradient descent, the weights matrix is subtracted from the previous weights multiplied by a learning rate. When the model converges, not only does logistic choose different words for positive and negative reviews, but it also ranks words that appear in both linear and logistic models differently. Some words in the z-score ranking do not appear in the logistic choices, but that is expected as there is a large amount of ameliorative and pejorative words in the final word selection and we use 100 words in total.

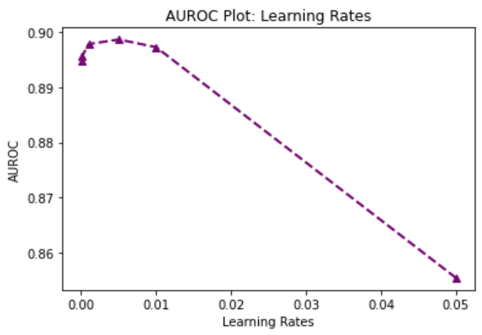
Logistic AUROC Plots



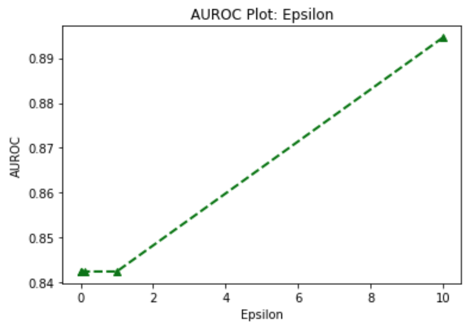
Compared to KNN, we see that Logistic regression yields a better AUROC score. At the optimal training size of 1, the Logistic model has 84% accuracy, while KNN has only 37%.



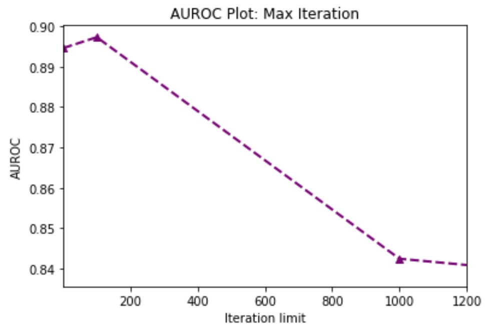
Logistic regression converges, with a CE of around 3.4. Through hyper-parameter testing, the AUROC of the logistic regression had a lower value of 0.84 to the highest value of 0.9.



The figure above displays another significant finding where learning rate could improve the AUROC of the model. Setting max iterations to 1000, the optimal learning rates ranged between 0.0 and 0.01.



An epsilon value of 10 resulted in the model's best performance whereas lower values result in lesser AUROC values. This could be explained by the model overfitting the training data.



Lastly, the highest AUROC achieved when limiting iterations is with a limit of around 150 iterations.

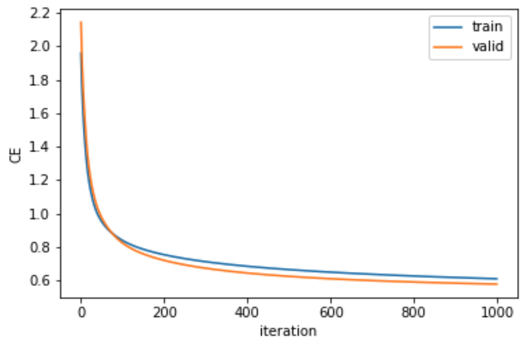
In sum, the regression's performance is affected by the choice of learning rates, epsilon, and maximum iterations, which demonstrates the need to optimize hyperparameters when seeking optimal accuracy. Overall gradient norm does decrease as those parameters vary accordingly, but sometimes too many iterations, or high epsilon values may decrease performance. The stopping criteria is a choice as it bets on the maximum iterations or other parameters to stop at the minimum gradient norm.

Ridge and LASSO

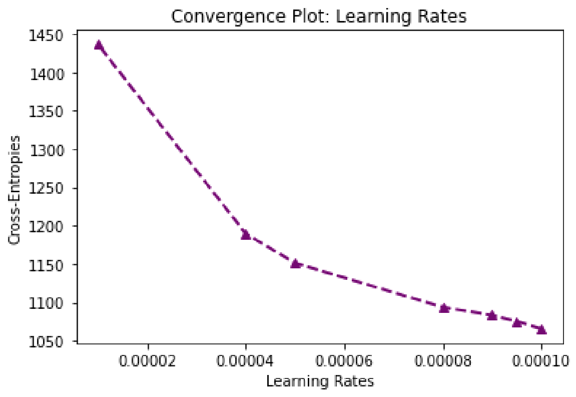
On the IMDB dataset, ridge regression performs equally well relative to the regular logistic technique. Varying for various alpha values also yields similar results, but no result is significant enough to make a meaningful statistical inference. LASSO, on the other hand, performs worse by a percentage point, which is noteworthy given the sample size that the model operates on. LASSO is used to eliminate the effect of hidden correlations between variables and as a result might be penalizing the logistic regression's power as it uses the negative and positive correlations.

Multiclass Model Results

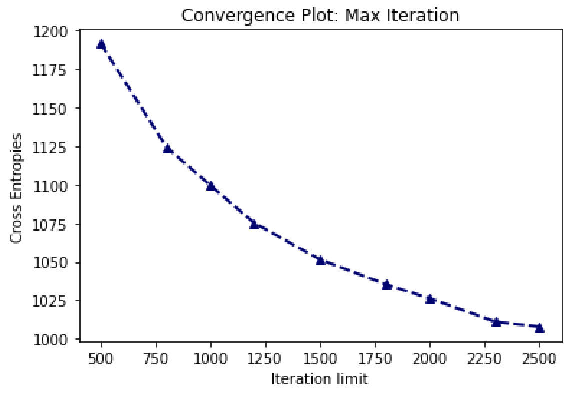
Running the basic multiclass regression test, we obtain an accuracy of 73%. The following plot represents the normalized cross entropy, at the different iterations, with a reasonable learning rate of 0.00005. We see that the normalized cross-entropy stabilizes at around 800 for both our training and validation dataset, which indicates that our model is not overfitting. This is validated by the fact that our training and validation accuracies are very similar (respectively 76% and 79%).



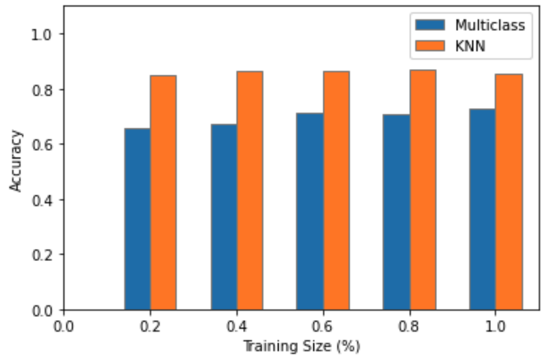
Next, we plotted the convergence of the cross-entropy, using the same learning rates as for the logistic. We see that the model starts converging at a learning rate around 0.00008, for a cross entropy around 1050. At this point, we also obtain an optimal training accuracy of 74%. Therefore, we used this learning rate in subsequent experiments.



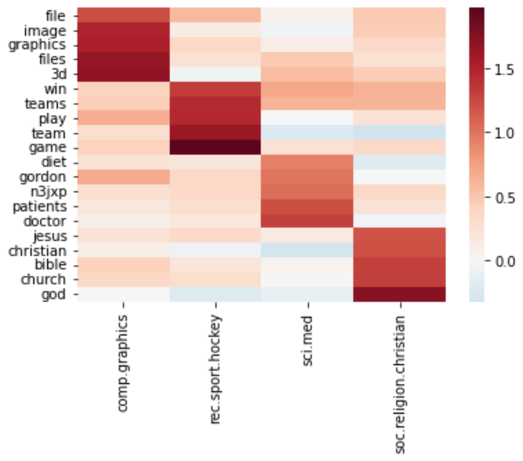
We experimented using different maximum iterations, from 500 to 2500. From the plot below, we see that the multiclass model starts converging around 2250, for a cross-entropy of 1000. Note that this is unnormalized CE, hence the high values.



We built a bar plot comparing the accuracies of the Multiclass regression and the KNN. We can see that KNN provides around 80% accuracy, for most test sizes, while Multiclass provides lower accuracy, at around 73%. A close inspection of the values indicates that more training does slightly improve performance.



Finally, we made a heat map of the weight for the 5 most positive features of our 4 news categories. We can notice that 5 words per news category appear more “red” than the other. This implies that each of those words are used to predict a category, which makes sense to attribute them more weight. This supports our model, as each of those words’ meanings are strongly associated with each category.



Discussion and Conclusions

Logistic Regression was accurate in predicting the classification of an article. As observed by the AUROC vs. hyperparameter plots, the regression can be improved through a validation process that chooses the adequate stopping criteria. This means that a model that runs longer may not always provide the best results or that an epsilon value may have to be increased to fit the data’s numerical structure. Ridge and LASSO regression did not offer significant upgrades to the regression as the data’s structure fits the statistical object of logistic regression. Also, some words are autocorrelated, which can be useful in predicting the correct article value.

Multiclass regression provides an accurate model to predict the classification of news articles, with an accuracy of 73% in its basic form. By changing the learning rate, the max iterations, and the test size, we can increase the test accuracy, as shown in our convergence plots. However, our experiments showed that the KNN model gives better prediction accuracy (87%). Furthermore, KNN involves no training. Therefore, we recommend using KNN instead of Multiclass Logistic Regression for this dataset.

While Logistic and Multiclass regression are not performed on the same dataset, we can still compare them and discuss some points. Our experimentation showed that Multiclass needs a high maximum iteration to reach its optimized cross entropy, compared to logistic. This makes the Multiclass model slower to train, as more iterations are needed.

With that said, further exploration could be performed. One extension, which was unfeasible for us due to limited computational resources, is a more extensive hyperparameter optimization by searching through a larger value space. Additionally, exploring alternate ways of handling words could also be interesting, for instance by using n-grams of variable sizes. Finally, exploring performance gains using more words per class for the multiclass could prove fruitful despite more intensive training.

Statement of Contributions

Adam worked on the 20 news group processing, the logistic regression exploration, and the introductory and analysis parts of the write-up. Frederic worked on the IMDB processing, multiclass regression exploration, and the results and discussion sections. Finally, Yann worked on the model implementations, the required base explorations for both models, and the latter portions of the write-up.

Bibliography

1. Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).*
2. *K. Albishre, M. Albathan and Y. Li, "Effective 20 Newsgroups Dataset Cleaning," 2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2015, pp. 98-101, doi: 10.1109/WI-IAT.2015.90.*