PROJECT 3: MOVIE REVIEW SENTIMENT ANALYSIS

Jonas Reger (wreger2), Hope Hunter (hhunter3)

November 29, 2022

1. Team Member Details

Our team was composed of two members, Jonas Reger (wreger2) and Hope Hunter (hhunter3). Both members discussed and contributed to both the coding and writing of this report. Jonas mainly focused on developing the general framework from vocabulary selection to model fitting and predictions, and writing the documentation for vocabulary set building. Hope mainly focused on attempting alternative vocab construction methods and paper writing. Both members reviewed each other's contributions regularly and correlated them into final deliverables for the project.

2. Introduction

We are given a set of 50,000 IMDb movie reviews, where each review is given a binary label of positive or negative for sentiment score. This dataset was collected in association with the publication "Learning Word Vectors for Sentiment Analysis" [1]. Movie reviews (assigned a number on a 1-10 scale) which have an overall rating 1 through 4 are assigned a 0 and reviews which have a rating 7 through 10 are assigned a 1. These are divided into 25,000 test reviews and 25,000 train reviews. Five iterations of train-test splits are created from this data set. Our goal is to build a binary classification model which can predict whether a movie review has a positive or negative sentiment using a vocabulary set of highly interpretive terms. To do this, we must produce a vocabulary of 1000 or fewer terms and a classification model which can be used to predict movie review sentiment. The metric with which our model is evaluated is area under the Receiver Operating Characteristic (ROC) curve (AUC) which should be greater than or equal to 0.96 for all five splits. Achieving $AUC \geq 0.96$ for all splits means that the model correctly classifies at least 96% of the reviews in the test data.

3. Pre-processing

We use the following files in our project:

- train.tsv: five folds with three columns ,"id", "sentiment", "review"
- test.tsv: five folds with two columns,"id", "review"
- test_y.tsv: five folds with three columns, "id", "sentiment,"score"
- myvocab.txt: the selected global vocabulary set under 1000 terms for predictive modeling

Only the text data, IDs, and sentiment labels associated with each review are used in this analysis. So, any other data was ignored. To clean the data for use, we first remove punctuation, html symbols, stopwords, and extra blanks from words. We also turned all words to lowercase and perform tokenization with the "itoken" function. A document-term matrix is constructed using n-grams (maximum of 4-grams) after pruning the tokenized vocabulary. Much of this code comes from the Campuswire posts from Dr. Liang. Additionally, the document-term matrices are normalized with TermFrequency-InverseDocumentFrequency (tf-idf) transformation. The document-term matrices for train-test data are then conformed to the global vocabulary set (i.e. the column space matches the vocabulary set). Note that the pre-processing is very similar for the vocabulary set selection and predictive modeling procedures. A minor difference is the vocabulary set selection procedure does not use document-term matrix column conformation to the vocabulary in `myvocab.txt`, since it has not been created yet. Instead, the document-term matrix is updated several

times with column conformation as the vocabulary set is reduced across several steps. Further details are discussed in the vocabulary building documentation.

4. Vocab construction

The document-term matrix of the full data is fitted with a LASSO regression model to select the $N_0 = 10,000$ most important terms in the data. Since the goal of this analysis is to select important terms that are easier to interpret, a simple screening method is used. A two-sample t-test is used to find the best $N_1 = 2,000$ terms to interpret. For purposes of ensuring the document-term matrices are compatible across vocabulary selection and predictive modeling procedures under a given vocabulary set, a helper function is used to conform the matrices to the vocabulary set. After selecting the best terms via LASSO selection and the t-test screening method, LASSO is re-applied on the reduced document-term matrix to select the global vocabulary set within the target size, $N_T = 1,000$. Our final vocab selection has 994 words.

5. Model

The objective of the predictive modeling procedure is to predict the probability that a review is positive (the probability of negative reviews is implied as reviews are binary). So, if the vocabulary constrained document-term matrices contain useful information, then the model should correctly classify at least 96% of the test reviews with that as the data input. Since the response data is binary (0-1), logistic regression will be used. The focus of the predictive modeling procedure is not on feature selection. Instead, the goal here is to use shrinkage (Ridge).

We created a generalized linear model with the following parameters:

- Train data (predictors): the vocabulary-constrained document-term matrix of the train reviews (i.e., `x=dtm_matrix`)
- Train data (response): the sentiment labels of train reviews (i.e., `y=train$sentiment`)
- Cross-validation loss measure: the loss to use for cross-validation (i.e. `type.measure=deviance`)
- Number of folds: the number cross-validation folds (i.e., `nfolds=5`)
- Distribution family: logistic regression (i.e., `family=binomial`)
- Regularization term: ridge regression (i.e., `alpha=0`)
- Convergence threshold: change of loss stopping criterion (i.e., `thresh=1e-5`)
- Maximum iterations: iteration stopping criterion (i.e., `maxit=1e3`)

The penalty parameter used in the prediction function is the lambda with minimum Mean Squared Error (MSE) (i.e., `lambda.min`). This model was trained on the train document-term matrix that was conformed to our 994 word vocabulary list. Since we had already applied LASSO selection and the t-test screening method, the selected vocabulary set was essentially a set of features with suppressed noise and highly informative for interpretative and predictive purposes. Thus, it was justifiable to use a lower maximum number of iterations (`maxit=1e3`) and a higher convergence threshold (`thresh=1e-5`) in the cross-validated Ridge model compared to default parameters of the stopping criteria (`maxit=1e05, thresh=1e-10`). This was beneficial towards greatly reducing processing time (approximately 60-80% less time). The final output of the model is a list of predicted probabilities that each review is positive.

6. Results & Specs

We used a Windows 10 PC, which has 3.6 GHz Intel i7-9700K CPU with 32 GB of RAM to handle all procedures related to this analysis with a total run time of 31.54 minutes (7.94 minutes for vocabulary selection and 23.60 minutes for modeling). Below are the error metrics (AUC) and run times (Seconds) for each train-test split.

Train-Test Results

Split	1	2	3	4	5	Mean
AUC	0.96816	0.96730	0.96741	0.96773	0.96788	0.96770
Time	283.190	274.629	288.105	277.662	292.804	283.278

As we can see, the AUC reaches the benchmark of 0.96 for all five splits.

7. Discussion & Conclusion

While it may not be entirely appropriate to build a global vocabulary over the entire data, there were two benefits that support this approach. First, we saw significant improvements in processing time, reduced by approximately 30-40 minutes. Secondly, we treated the vocabulary selection as an analytical procedure to separate selecting interpretive terms for the vocabulary set, and predictive terms for the predictive model. This distinction gave us justification for this approach of building a vocabulary set from the entire data. Additionally, conforming document-term matrices between train-test data to the vocabulary set enables more focus on the predictive aspect in the prediction model. A possible modification of the vocabulary selection procedure is to update the vocabulary over time as new data is observed. This would make the approach more appropriate while also having lower computational expense. Regardless, it was very interesting to attempt to distinguish and balance interpretive and predictive inferential procedures. Despite the theoretical limitations of using a vocabulary set extracted from the entire data, the model performed very similarly even when the vocabulary was extracted only from train data. So, hardly any predictive or interpretive power was lost while computational expense was lowered significantly (nearly 80% reduction in run time overall).

While the model performs quite well with accurate classifications for over 96% of reviews, there are sources of errors. First, natural language is complex and oftentimes difficult to quantify accurately. Thus, one likely source of errors are terms with highly ambiguous meanings. If a word is significant but has ambiguous meanings for both positive and negative sentiments, then it could be filtered out by the t-test screening method. Neural networks may be more suited for capturing word contexts that are useful for inference in this situation. Second, our procedure focuses on extracting relevant information at the word or phrase level using a simple document-term matrix with regularized logistic regression. Neural network models such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) have been shown to generalize contextual meanings around words better than traditional methods. This may allow development of new screening procedures that select more informative contexts while simultaneously handling ambiguities. The challenge of using neural networks is to select highly interpretive features for the predictive model, since networks are typically more black-box than white-box. Perhaps, even a topic modeling procedure may be useful for evaluating this. There is a BERT Topic transformer model[2] that is used to cluster topics with the transformer, then uses tf-idf to pick out most representative terms of each cluster. Something similar could be adapted in combination with the two-sample t-test procedure to select better contextual features.

The main limitation of the model is that it is unable to generalize contextual meaning very well, since it is a bag-of-words approach that does not consider word-order or other linguistic features. Neural networks would be able to generalize better in comparison and may improve model performance when new data is observed. Traditional bag-of-words methods will not be able to extract meaning very well when unknown terms appear, but neural networks can. Another limitation of the model may lie in the sentiment scoring criteria. It is possible that certain terms may follow some unknown probability distribution of ratings (1-10). For instance, a term may follow a skewed binomial distribution that favors negative ratings, but is mostly neutral ratings. This could misinform the model that the term should be negative, which could cause misclassification of true positive reviews that contain the term (true neutral term). Perhaps, having access to the full scope of ratings may allow better understanding of how the distributions overlap, which could allow screening modifications to ignore neutral words. Lastly, a limitation of the model is in the fundamental construct of using a limited term vocabulary as predictors. Using topic models in combination with the document-term matrix and vocabulary set would help reduce the dimensions of the vocabulary set. This would make it possible to achieve similar classification accuracy with a much smaller vocabulary.

Interpretation Many terms selected in the final vocabulary set are intuitively informative to us as humans (e.g., bad, worst, great, waste, awful, excellent, etc). However, some words still seem a little bit less interpretative (e.g., also, very, just, one_of, 1, only, today, etc). This is the fallacy of bag-of-words methods, since word order and contextual meanings are not well generalized. Using neural networks with topic modeling in some combination with our methods, may yield improved representations of the vocabulary that distinguish or combine terms according to contextual meanings. Distinguishing ambiguities can help separate meanings that should be separated. Clustering similarities can help combine meanings that should be combined. Both of these processes could help simplify the meaning representations while also enriching them. In any case, despite the limitations of the framework towards enhancing interpretation of vocabulary terms, it still performs fairly well. The model simply responds to what terms a review has from the vocabulary set (the row of a document-term matrix contains the information of what terms in the vocabulary set that the review has). Since this is a Ridge model, shrinkage was applied to the coefficient estimates to optimize prediction performance. The model simply predicts higher probabilities for positive sentiment when positive terms have a greater presence than negative terms in the review, and vice-versa for negative sentiments. This is a small oversimplification, but still essentially how the model makes predictions given the data.

References

[1] Daly R. Pham P. Huang D. Ng A. Maas, A. and C. Potts. Learning word vectors for sentiment analysis. *The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)*, 2011.

[2] Maarten Grootendorst. Bertopic: Neural topic modeling with a class-based tf-idf procedure. *arXiv preprint arXiv:2203.05794*, 2022.