Stat542_Project3_Report

xxxx

11/29/2021

Introduction

The goal of this sentiment analysis is to classify IMDB movie reviews as either positive or negative.The dataset has 50,000 movie reviews. The performance of the model is evaluated on five different splits of the 50,000 IMDB movie reviews. The performance metric we are using in this project is Area Under the Curve (AUC). We also aim to build a classification model that is interpretable such that how the model works can be easily understood and the results can be easily interpreted in substantive terms.

Data Processing

The whole dataset, alldata.tsv, has 50,000 rows and 4 columns:

- id, the unique identification number;
- sentiment, 0 for negative, and 1 for positive;
- score, a 10-point score assigned by reviewers. Scores 1-4 correspond to a negative sentiment while scores 7-10 correspond to a positive sentiment. There are no scores or values of 5 or 6 for this variable.
- review, the actual text content of each review.

The 50,000 reviews are used to generate 5 sets of training/test splits, each containing 25,00 reviews. For reasons discussed in more details in the corresponding Rmarkdown file, we use the full 50,000 movie reviews to build our sentiment analysis model as it's very difficult to reach the desired benchmark level performance threshold when training the model on only one split of the data.

Before taking any formal model building steps, we have to perform a series of data processing to create the input data required for a text classification model for sentiment analysis. And most importantly, we need to create a desirable vocabulary list of positive and negative words that can be used to train the model to accurately classify the movie reviews.

While more details are provided in the attached Rmarkdown file, we briefly discussed the main actions taken to create the vocabulary list. First thing we did is to remove the HTML tags (possibly automatically generated as the movie reviews were submitted online). Next we take several steps to create the document term matrix (perhaps can be more precisely called review term matrix in this specific use case). These steps include tokenizing the reviews, filtering out commonly used stop words, including terms of minimum 1 word and maximum 4 words, including only terms that appear at least 10 times across the movie reviews and appear in at least 0.1 percent of the reviews and at most 50 percent of the reviews.

The resulting document term matrix created from previous steps is pretty large. In fact, it has more than 30,000 terms. To both avoid overfitting and make the trained model more interpretable, we need to substantially trim this large initial vocabulary list. The approach we take here is to fit a Lasso logistic regression model and in the corresponding model fit output, we pick the set of estimated term coefficients

with the largest degree of freedom less than 1000. This happens to be the 36th set with 982 non-zero term coefficients and therefore 982 terms. This list of 982 terms is our vocabulary list that is used to train the movie review classification model for sentiment analysis.

Model Training

While training the model, for each of the five splits of the data, we also remove the HTML tags and tokenize the movie reviews. And we use the vocabulary list created in previous steps to create the document term matrix. Notice that the term here doesn't mean one word, in fact it could be two words or multiple words. The vocabulary list we created in previous steps has terms of up to four words. Here in training the model, we only include one-gram and two-gram terms.

After experimentation with different model specifications, the model we settle on to classify the IMDB movie review based on language terms used in the reviews is cross-validated Lasso (logistic regression). As we have only two classes: positive and negative, the loss we use for cross-validation is AUC, that is area under the ROC curve.

The functional form of Lasso is shown below. Basically the 982 terms in the document term matrix are our features used to predict the sentiment of each movie review. And each of these term features has a corresponding coefficient β as shown in the following equation. The Lasso regression also includes a penalty term which is the L1 norm $\Sigma_{j=1}^{p}|\beta_j|$ with shrinkage parameter λ. The value of λ is chosen by cross validation. In our case, we chose the λ value that achieves minimum cross validated loss to make predictions and classify movie reviews in the test set.

$$\Sigma_{i=1}^{n}(y_i - \beta_0 - \Sigma_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\Sigma_{j=1}^{p}|\beta_j|.$$

Model Evaluation and Performance

We also create the document term matrix for the test data using the same steps discussed above (for training data). And we use the Lasso model trained in previous step to classify movie reviews in the test set as either positive or negative with the λ value chosen by cross validation. The model performance on the 5 different data splits are shown in the following Table 1.

For a classification task, the ROC curve (receiver operating characteristic curve) plots the false positive rate at the X-axis and the true positive rate at the Y-axis. AUC (Area under the ROC Curve) provides an aggregate measure of classification performance across all possible thresholds. One interpretation of AUC is that it measures the ability of the classification model to rank the positive case higher than the negative case, thus maximizing the true positive rate while minimizing the false positive rate. An AUC value closer to 1 is desirable. For all five splits of the data in this project, we are able to achieve above the benchmark level of AUC 0.96. More specifically, as shown in the following Table 1, the AUC values for movie review classification for the 5 data splits are 0.9664, 0.9668, 0.9673, and 0.9659 respectively. AUC is both scale-invariant and classification-threshold-invariant.

The model training and prediction was run on a Windows machine with Intel(R) Core(TM) i5-7300U CPU 2.60GHz and 8.00 GB (7.87 GB usable) memory, and the total run time for each split of the data ranges from 47.47 seconds to 49.57 seconds.

Discussion and Conclusions

Interpretability

We have trained a model that is easy to interpret. That is we are basically using terms that are most likely to reflect positive or negative sentiment to predict the overall sentiment of a IMDB movie review. As shown

Split	AUC	Processing_time_in_seconds
1	0.9664	48.2999
2	0.9668	49.3556
3	0.9664	48.8200
4	0.9673	49.5722
5	0.9659	47.4743

Table 1: 5 Fold Model Performance and Processing Time

in the following Table 2, the vast majority of the top 25 positive and negative terms can be easily seen to reflect either a strong positive or a strong negative sentiment, thus they are good indicators of the overall movie review sentiment and good terms to be included in the movie review sentiment classification model. The corresponding beta values point to the magnitude of their contribution to the overall movie review sentiment classification.

Top_Positive_Words	Positive_betas	Top_Negative_Words	Negative_betas
must_see	0.82	waste	-1.21
excellent	0.67	worst	-1.04
superb	0.59	awful	-0.88
wonderful	0.51	poorly	-0.79
amazing	0.50	terrible	-0.60
beautifully	0.49	boring	-0.60
favorite	0.47	worse	-0.52
perfect	0.47	horrible	-0.50
brilliant	0.41	poor	-0.50
fantastic	0.41	ridiculous	-0.45
great	0.39	bad	-0.41
loved	0.34	avoid	-0.40
enjoyed	0.30	supposed	-0.35
best	0.26	stupid	-0.35
highly	0.21	crap	-0.32
beautiful	0.19	nothing	-0.32
both	0.14	minutes	-0.27
love	0.13	at_all	-0.26
life	0.11	why	-0.15
always	0.11	so_bad	-0.13
well	0.10	plot	-0.11
also	0.08	no	-0.10
very	0.06	acting	-0.08
one_of	0.05	even	-0.06
performance	0.01	just	-0.05

Table 2: Top 25 Positive and Negative Words

Model Limitations

While the model we trained in this project successfully reached the desired benchmark performance level for all 5 splits of the data, it is not without its limitations. For one thing, logistic regression is known to have a rigid decision boundary, that is it has a fixed cutoff value of 0.5. Difference between calculated probabilities either side of 0.5 doesn't have an impact on the final classification no matter how different they are. For example, 0.51 and 0.99 would have the same classification even though one is only marginally positive while the other one is extremely positive. If we want to do more fine-grained sentiment classification of multiple classes, then we probably need to choose a different type of model.

In addition, there exist cases where incorrect classification was made by the model we trained. Take the first split of the data as an example, there is about 1 percent miss-classification rate[1]. We wrongly classified 1067 positive movie reviews as negative and also wrongly classified 1265 negative movie reviews as positive. For example, the following review 52 was wrongly classified as being positive while in fact it is negative probably because of the positive terms embedded within the text, for example "enjoy", "inspiring", and "feel_good". To understand the context in which these positive words are used and how they are getting misinterpreted by the model we trained is a necessary next step for improving our sentiment classification model.

Review 52: "*As a serious marathoner, I was seriously disappointed in this film. Its target audience is clearly those who have never run a marathon, or novice marathoners. Following the stories of 2 first-time marathoners, one senior, one injured runner, and two elites as they prepare for the Chicago marathon, the film dedicates the majority of its attention to one female beginner whose story is, for lack of a better word, boring. While I did enjoy the brief glimpses into the training sessions of Deena Kastor, the brief history of the Boston marathon and marathoning in general, let me emphasize: These were brief!! Watching some Joe Runners prepare for a Saturday run with their water bottles and talking about how they view the marathon is not inspiring, and the nonstop cliches about achievement and feel-good grinning runners will make you wish the film were about an hour shorter. If you are a first-time marathoner, this film may give you a feeling of "I can do it." For anyone else, run away.*".

Future Steps

There are many areas to improve upon for the movie review classification model we constructed in this project. For one thing, we could have constructed the document term matrix in different ways, for example we might use the Term Frequency Inverse Document Frequency (TFIDF) method which could potentially give us better model performance.

Related to the classification errors mentioned above, we could have adopted more sophisticated methods to better account for the contexts where certain words or combination of words are used. For example, positive terms could be used in a more broadly negative context and negative terms could be used in a more broadly positive context, both cases are scenarios likely to produce miss-classification. Our next stage model building could attempt to consider terms that can qualify the sentiment of a context within which terms appear in text. More advanced methods, for example bidirectional Long Short-Term Memory, might be tools that could be useful.

We also could certainly construct a more comprehensive list of stop words as that will help remove words that are very unlikely to contribute to the sentiment classification on the movie reviews.

Acknowledgement

Related course materials greatly helped the smooth implementation of the model in this project.

[1]There might be an overfitting issue as well.