Darrell Nelson II

Text Mining (IST - 736)

08/30/2019

HW07: MNB and SVMs for Sentiment Classification

Introduction

Sentiment classification is the process of determining the emotional tone behind a series of words, used to gain an understanding of the attitudes, opinions, and emotions expressed within text. It is generally deployed by companies to better predict what their customers and potential customers think about a certain topic. Sentiment analysis is extremely useful in monitoring social media as it creates a synopsis/deep overview of the wider public opinion behind a topic.

With the help of sentiment analysis systems, the unstructured textual data can be automatically transformed into structured data of public opinions about products, services, brands, politics, or any topic that people can express opinions about. This data is very useful for commercial applications like marketing analysis, public relations, product reviews, net promoter scoring, product feedback, and customer service. The applications of sentiment analysis are broad and powerful. The ability to extract insights from social data is a practice that is being widely adopted by organizations across the world.

Analysis and Models

About the Data

Dataset is comprised of tab-separated files (TSV) with phrases from the Rotten Tomatoes dataset. The train/test split has been preserved for the purposes of benchmarking, but the sentences have been shuffled from their original order. Each Sentence has been parsed into many phrases by the Stanford parser. Phrases and sentences have their own unique PhraseId and SentenceId. Phrases that are repeated (such as short/common words) are only included once in the data (Kaggle 2019). Dataset contains 66,292 parsed dialogues.

- ‘train.tsv’ contains the phrases and their associated sentiment labels.
- ‘test.tsv’ contains just phrases.

The sentiment labels are:

0 - negative
1 - somewhat negative
2 - neutral
3 - somewhat positive
4 – positive

```
PhraseId   SentenceId Phrase     Sentiment
1     1     A series of escapades demonstrating the adage that
what is good for the goose is also good for the gander , some of
which occasionally amuses but none of which amounts to much of a
story .     1
2     1     A series of escapades demonstrating the adage that
what is good for the goose  2
3     1     A series    2
4     1     A     2
5     1     series      2
6     1     of escapades demonstrating the adage that what is good
for the goose    2
```

Figure 1. Raw TSV file

The raw TSV file reviews were imported into a python IDE (Jupyter Notebook) and the column 'Phrase' was saved as the input (X) and the column 'Sentiment' as the output (y) variable. Data was then randomized and separated into training/testing sets with a 60/40 split respectively. Initial analysis of the training data shows that the dataset is skewed towards "neutral" examples with ~51% of all the observations in the training set being neutral. A good model is defined by having a higher probability than random chance, and properly addressing skewed data without bias towards the skewed output. It must properly address all classes without highly relying on "default-guessing" the skewed class. Models in this experiment were checked to ensure they have a higher predicted accuracy than if the model just randomly picked the answers to be part of the skewed class.

Models

Multinomial Naïve-Bayes

The multinomial naïve-bayes model is an application of supervised machine learning. It employs a learning algorithm to identify a model that best fits the relationship between the attribute set and class label of the input data. Then, uses joint and conditional probabilities to draw relationships between inputs and outputs. A Naïve-Bayes classifier estimates the class-conditional probability by assuming that the attributes are conditionally independent, given the class label. This assumption is violated in a lot of real-world cases, especially in text mining. Based off syntax and semantics in human language only certain words are followed after others in a sentence, especially if they carry a lot of meaning in that

sentence. Although there are a variety of different words one can use, the proceeding words are determined by the language in which the texts are constructed.

Support Vector Machine

Support vector machines (SVM) are a classification technique that is used to run supervised machine learning. This model has its roots in statistical learning theory and has shown promising empirical results in many practical applications, including handwritten digit recognition and sentiment analysis. SVMs also work well with high-dimensional data and avoid the curse of [the] dimensionality problem. Another unique aspect of this approach is that it represents the decision boundary using a subset of the training examples, known as the support vectors.[1] The decision boundary is chosen by maximizing the distance between datapoints within a cluster and the boundary itself without sacrificing accuracy. This increases the chance of properly labeling unknown test data that may reside near the boundaries. A linear test boundary was tested during this experiment with different cost parameters that penalize the model for misclassifications.

Results

The 'Phrase' training data was vectorized, stop-words removed, and all words that did not appear in a minimum of 5 phrases were removed. The vectorized data was trained on a multinomial naïve-bayes (MNB) classifier. In order to ensure that the model is "learning" the vocabulary, the features (words) with the highest weight produced by the model are output for the most positive and most negative rankings *(Figure 2.)*

```
Top 10 most negative words in MNB:
 [(1.5210602737939296, 'waste'), (1.523562078883295, 'minutes'), (1.533975977581715, 'poorly'), (1.5402375494766518, 'awfu
l'), (1.5463767818167982, 'contrived'), (1.5463767818167982, 'unfunny'), (1.5698200666534547, 'worse'), (1.5969581857419417,
'stupid'), (1.7858251634181175, 'worst'), (1.857793692114055, 'bad')]
Top 10 most positive words in MNB:
 [(0.5958791031947216, 'moving'), (0.604612353536404, 'beautiful'), (0.6156827739590933, 'beautifully'), (0.621740385324800
4, 'powerful'), (0.6249198570600717, 'solid'), (0.6298970899440408, 'touching'), (0.6333656656276786, 'gorgeous'), (0.636965
1614685079, 'excellent'), (0.6420878254266654, 'best'), (0.6528979981733687, 'wonderful')]
```

Figure 2. Top 10 most negative and most positive words in MNB model

Words like *waste, poorly*, and *awful* made it to the top of the 'most negative' list while *moving*, *beautiful,* and *powerful* were among the best in the most positive list.

[1] (Tan, Steinbach, & Kumar, 2006)

The model was then tested with the testing dataset and precision, recall, and F1 scores were calculated:

```
[0.45689655 0.49734812 0.67337708 0.51132554 0.482231  ]
[0.25315592 0.38118995 0.80831032 0.47849709 0.26143966]
              precision    recall  f1-score   support

           0       0.46      0.25      0.33      2931
           1       0.50      0.38      0.43     10824
           2       0.67      0.81      0.73     31864
           3       0.51      0.48      0.49     13068
           4       0.48      0.26      0.34      3737

   micro avg       0.61      0.61      0.61     62424
   macro avg       0.52      0.44      0.47     62424
weighted avg       0.59      0.61      0.59     62424
```

Table 1. Performance Measures MNB

The model did very well in terms of class precision. About half of all the predicted values in each class were correct; class 2 is the outlier with a 67% precision which is due to the unbalanced nature of this dataset. Class 2 holds ~51% of all training/testing cases therefore its higher performance must be taken with a grain of salt. Recall appears to be the best metric to perceive model performance in this experiment. Recall calculates how many actual positives were calculated from the predicted positives. This metric penalizes the model for guessing based on skew. The model struggled with the most extreme classes ('very negative' and 'very positive'), but as the phrases got more neutral the model's precision increased. F1 Score is a type of average that equally weights the importance of precision and recall and outputs it as a single number.

Micro- and macro-averages compute slightly different averages, and thus their interpretations differ. Macro-average computes the average independently for each class and then takes the average (treating all classes equally), whereas a micro-average will aggregate the contributions of all classes to compute the average metric. In a multi-class classification such as this, micro-average is preferable due to the class imbalance. Therefore, the main grading criteria for model comparison in this instance will be the micro-average recall metric which is at 61% for the MNB model.

The vectorized data was also trained and tested on a Linear SVC (Support Vector Classifier). This support vector machine (SVM) model appears to have a better understanding of sentiment based on its top 10 features in the 'very negative' and 'very positive' classes. Words like *cesspool, disappointment*, and

pompous pop out as very distinctive words that have extremely negative connotations. On the positive side words like *stunning*, *astonish*, and *refresh* are all distinctively positive. (*Figure 3.*)

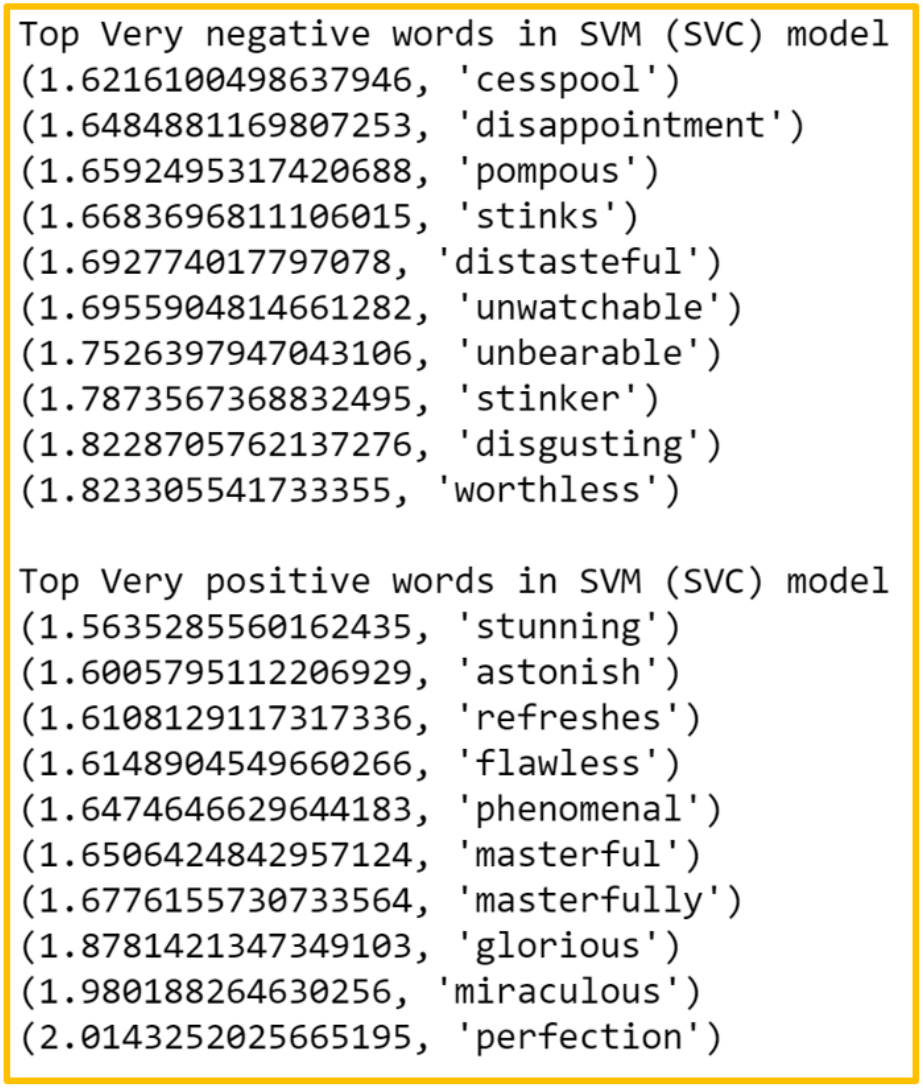

Figure 3. Top 10 most negative and most positive words in SVM model

The higher resolution vocabulary words should make for a more accurate model!

	precision	recall	f1-score	support
0	0.50	0.31	0.38	2931
1	0.52	0.38	0.44	10824
2	0.68	0.85	0.75	31864
3	0.54	0.42	0.48	13068
4	0.51	0.35	0.42	3737
micro avg	0.62	0.62	0.62	62424
macro avg	0.55	0.46	0.49	62424
weighted avg	0.60	0.62	0.60	62424

Table 2. Performance Measures SVM

Viewing the micro-average recall there is a 1% increase from the previous model. With a testing set of over 62,000 samples, that's an improvement of over 6,000 samples! A deeper dive into the recall column shows that this improvement is due to the improved accuracies at the extremes. In the naïve-bayes model the recall for the 'most negative' and 'most positive' classes were 24% and 25% respectively. In the SVC those classes had a recall of 31% and 35%.

Overall, the SVM yielded ~2% increase in accuracy over the MNB.

```
MNB accuracy:  0.606401384083045
SVM (SVC) accuracy: 0.6236864026656415
```

MNB vs SVM Accuracy

Bigram vectorization was also implemented to improve micro-average recall and overall accuracy:

Bigram Vectorization

	precision	recall	f1-score	support
0	0.40	0.30	0.34	2931
1	0.48	0.41	0.44	10824
2	0.68	0.77	0.72	31864
3	0.51	0.49	0.50	13068
4	0.43	0.31	0.36	3737
micro avg	0.60	0.60	0.60	62424
macro avg	0.50	0.45	0.47	62424
weighted avg	0.58	0.60	0.59	62424

MNB Bigram

	precision	recall	f1-score	support
0	0.47	0.35	0.41	2931
1	0.52	0.42	0.47	10824
2	0.70	0.82	0.76	31864
3	0.55	0.46	0.50	13068
4	0.49	0.39	0.44	3737
micro avg	0.63	0.63	0.63	62424
macro avg	0.55	0.49	0.51	62424
weighted avg	0.61	0.63	0.62	62424

SVM Accuracy

```
MNB accuracy:  0.5973824170190952
SVM (SVC) accuracy: 0.6300941945405614
```

MNB vs SVM Accuracy w/Bigram Vectorization

Bigram vectorization decreased the MNB model's recall micro-average by 1% while improving the SVM's by 3%. This pattern is also reflected in the overall accuracies. While intuitively adding bigrams into a MNB model should improve accuracy as the model assumes word to word independence (the proceeding word has nothing to do with the word before it). This is not the case in the structure of the English language, however the MNB results did not reflect that. A deeper dive into the mechanics of the MNB model is required to understand why adding bigrams would result in a worse performance.

Lastly, a new SVC model was trained using all of the available data as its training source and its training accuracy was computed using 10-fold cross validation. The values in the features were calculated by using inverse document frequency; all stop-words and words that did not show up in at least 5 of the documents were removed. The model achieved an overall accuracy of ~81%, the highest achieved in this experiment. A few of the top features for the 'most negative' class were: *pathetic*, *basketball teams*, and *utterly incompetent*. The first and last word(s) are very demeaning and it's understandable that the model would properly rate them as extremely negative. However, *basketball teams* seem out of place. Perhaps the movie reviewers simply don't like cliché sports movies?

Some of the most positive words were: *masterful*, *glorious*, and *flawless*. (*Figure 4.*)

```
Top Very negative words/word pairs in SVM (SVC) model
(3.2082422129261308, 'pathetic')
(3.2215280338111936, 'basketball teams')
(3.228181220179103, 'utterly incompetent')
(3.271839748666288, 'paper bag')
(3.3126557201266467, 'unbearable')
(3.322777143463517, 'movie contrived')
(3.3531562605706466, 'Skip')
(3.3948479865391876, 'movie titled')
(3.5915721464868517, 'disappointment')
(3.8838824149766777, 'admit walked')

Top Very positive words/word pairs in SVM (SVC) model
(1.651780329327719, 'masterful')
(1.6642264040041055, 'glorious')
(1.6946065602147025, 'flawless')
(1.7364395750683885, 'masterfully')
(1.738277853018173, 'gem')
(1.744519740734703, 'miraculous')
(1.8078519838505431, 'cut rest')
(1.8597827705487435, 'amazing')
(2.022840068620278, 'masterpiece')
(2.1269100417311484, 'perfection')
```

Figure 4. Top 10 most negative and most positive words in optimized SVM model

Conclusion

Using MNB or SVC models to predict sentiment proved to be more reliable than random guessing (20%) and guessing based on skew (51%). Therefore, both models have validity and can assist in predicting sentiment.

The use of different vectorization methods can have a great effect on the accuracy of a model. Although, more complex/robust does not always lead to improved performance. For example, term frequency-inverse document matrix appears to be the best vectorizer used in this experiment because it accounts for word frequencies between documents and assigns them a normalized weighting. However, it didn't perform better than the generic baseline vectorizer with the multinomial model. This phenomenon is what makes Data Science an art as much as a science.

In conclusion, the SVC model trained on bigrams using inverse document frequency proved to be the highest performing model in this text mining classification. Language is a lot more complicated than 1s and 0s and in order to build the best predictive models more effort needs to be put into understanding the baseline data and transforming sentences and words into corresponding semantic values. So much of the performance of a model is based on the preparatory side and understanding how the model is interacting with the data and how it is creating relationships. The best models are created before they are even trained.