Homework 4

Kyle Walter

11/4/2021

# Introduction

Like most modern societies it is becoming more and more common to eat out in America. As the younger generation feigns cooking for more restraunt curated meals, the food industry even through the pandemic continues to see a bit a of renissance in options, even in very small often hard to reach places.

More and more as social media has entered the market, people have used the medium to impart on the public infomration about various products. Amazon the world’s largest online retailer, amoung other digital services, leads all advertisements with the overall rating form previous buyers. It also hounds those who buy a product to leave a recommendation to help future buyers.

As consumers have become more accustomed to this over the years, it take almost no no leap in imagination to see that people would be interested in similar review systems for restaurants. Unlike Amazon however; restaurants often are family owned, with limited or very dated web interfaces, and the ability of the owners to take a vacation, let alone program a review site, is often hindered by the demands of running the operations.

Thankfully there are services like Yelp, that seek to collect informaiton about various restaurants in an area and inform the public. It does so through reviews, however; unlike Amazon there is no “verified” diner meaning anyone with grudge could start a chain of negative reviews driving future customers making decisions off of these services to the competition. This can have devestating impacts on the restaurants bottom line and potentiall cost those who work at the restaurant their livelihood in lost wages or jobs from the customers who never show thanks to reviews. Restaurants could also game the system by leaving fake positive reviews, driving business to them that leaves people not wanting to return and at the same time hurt others that were deserving of the business with real reviews.

Can data science help determine if the reviews left by customers on these sites are real as well as detecting when a negative review is posted so the staff is able to quickly identify and hopefully remdy the situation?

# Analysis

For this project a data set of 92 reviews have been provided from the website. The Reviews are labeled first with a category indicating if they are lie and the second category indicating if the review is positive or negative.

The data was provide both in a comma separated format and a tab separated format. In the comma separated version the reviews which also have commas created some issues in python. Thankfully, since the body was all in the last column, a quick tweak of the pandas.read\_csv function allowed the data to brought in 3 columns correct.

In the initial data displace, it was noted that ’ signs typically noting contractions or possessives in English all had a "" mark following them. These were removed as a starting step. Followed by converting all words to lower case.

Next the words were converted to Tokens using Natural Language Tool Kit’s word tokenizer. A token is a unit that a computer can use to count. Typically practice is to capture words or short phrases for the computer to count. The following step was to lemmatize the words. Lemmatization is conversion of words with morphological changes back to what is generally considered a recognizable root. Example “walks” to “walk”. It generally does not change the meaning but helps reduce the number of features for the later step of building a model as more features typically will increase the time it takes a model converge on a result. In addition too many infections can cause patterns to get lost and model to preform less optimally, but there maybe times where these changes could negatively impact the model.

The next pre-processing step was to remove the stop words, or words that add to the structure of a sentence, but often time do not provide real meaning. Examples from english are words like I, you, the, a and so on. NLTK has a built in list of English stopwords that worked as the starting point. From prior projects, it has been recognized that some stop words like not, will impact the ability of the models learn sentiment. Examples like “This restaurant can not get orders out on time” to “This restaurant can get orders out on time” takes the sentence from something that reads as negative to neutral or maybe positive depending upon the reader. Since one of the goal is to give the model the best ability to recognize sentiment, these stop words were removed from the stopword list prior to the words removal from the main reviews.

The last analytically step before model building was to look at the class balances for the categories the model is seeking to learn. In both the cases of lies and sentiment the class were equally split at 46 observations of lies and truths, as well as 46 positive and negative sentiments each. Class balancing is important in the fact that it allows model to learn patterns associated with each classification. Where as unbalanced sets leads a model to learn the more dominate class as a default making the model more likely to choose the dominate outcome more often (or always) skewing the results.

## f 46  
## t 46  
## Name: lie, dtype: int64

**Lie class balance**

## n 46  
## p 46  
## Name: sentiment, dtype: int64

**Sentiment Class Balance**

## Vectroization

The final step of pre-processing is vectorization of text. This takes the tokens and creates a matrix like array of all the Tokens in the Corpora and provides so numeric value for the model to use in the next step for learning how to apply the classifiers. Sklearn in python provides a few different vectorizers to complete this task quickly and easily.

This stop along with the model in the following section were actually tried a few different ways to return the best results for the model. The best route was found to be Term Frequency Inverse Document Frequency or tfidf. This type of vectorization is converts counts the frequency of the words in a given corpus and uses the inverse of occurrence to give words seen less frequently throughout the Corpora a higher score making them more prominent in the model.

Sklearn’s vectorization comes with built in tokenization, stopword Removal, and the ability to create not only unigrams but ngrams as well. It also allows for the specification of which Tokenizer to use and stopword list to use.

By searching online, code for combining both tokenizer and lemmantizer were found so that it can be past as one of the tokenizer argument into the Tfidf Vectorizer. The other measures that were defined include the stop word list designed earlier to leave in the negations and the us the use of both unigrams and bigrams so that negative helpers remain together with the words they are supporting.

# The Models

The final step is to create and review the models.This task is focused on one type of model in particular, the Naive Bays Multinomial Model.The Model assumes independence of the observation and when used in text mining it is known that language does have dependent structures however, the advantage of this model is that high number of dimensions do not drag out the computation time of the model. It is a supervised model meaning both the independt features and dependent features are inserted into the model building process for it to discover the relationships between the features.

The other advantage of Naive Bayes is that upon the build of the model, it is relatively easy to tease out the features that are most informative and least informative to the model which can help lead to better training results if returned and rerun.

For this particular task, the other model testing feature being utilized is cross-fold validation. When testing model output two common methods are used. First the hold out method which holds a certain % of records from being used for training they are then supplied for the model to make a prediction and compared against their result. The other is is cross fold validation where the entire dataset is used for training. In the background it split the training set into the desired number of sets and holds out 1 and then builds and test the model and repeats this process till all sets have been tested and the results can then be average to get a better sense of how the model preforms overall. This test will use 10 folds where 90% of the data is trained on each round and 10% is used for testing.

For Python first step is to pull out the variables used for independent variables. In this case the csv data was recalled and shuffled randomly using the the sample feature and sampling for all records.

## lie sentiment review  
## 48 f p 'in my favorite restaurant yuenan restaurant. ...  
## 86 t p 'blue monkey cafe is my favorite japanese rest...  
## 37 t n 'the food was not bad, but the place was all n...  
## 20 f n 'usually, i use yelp to find restaurant. the y...  
## 72 t p 'stronghearts cafe is the best! the owners hav...

**Sorted Data Frame**

This way the tags will appear in the model calculation in a random order. Next the the independent data was run through the Tfidf Vectorizer with the aforementioned tokenization steps and saved the a variable for passing to the sklearn model.

## # # winning $ ... yuenan restaurant ’ ’ isle  
## 0 0.000000 0.000000 0.000000 ... 0.219666 0.000000 0.000000  
## 1 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 2 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 3 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 4 0.119662 0.119662 0.097006 ... 0.000000 0.000000 0.000000  
## .. ... ... ... ... ... ... ...  
## 87 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 88 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 89 0.000000 0.000000 0.000000 ... 0.000000 0.111212 0.111212  
## 90 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
## 91 0.000000 0.000000 0.000000 ... 0.000000 0.000000 0.000000  
##   
## [92 rows x 4698 columns]

**TFIDF Matrix**

Following that the tags for lies and tags for sentiment were both saved in separate dependent variables.

The Variables were then passed into the Multinational model inside the cross\_valdation\_prediction with the 10 fold cross validation. The models prediction for lies and sentiment were saved in new variables and passed with the given tages into the classification report function of the sklearn model to see how well the model are preforming.

The model for lie after several tries, both with CountVectorizer and TFidfVectorizer showed the best result with the features and tokenization described above. The best however; is still not very good.

## precision recall f1-score support  
##   
## f 0.55 0.59 0.57 46  
## t 0.56 0.52 0.54 46  
##   
## accuracy 0.55 92  
## macro avg 0.55 0.55 0.55 92  
## weighted avg 0.55 0.55 0.55 92

**Classification Report for Lie Detection**

As seen in the results the model’s ability to detect deception is only at 55%, only slightly better than guessing. The model is able to recall lies it has been trained on slightly better at 59% but over all this model currently does not support the ability to detect deception.

The same parameters when used for sentiment however; did fair quite a bit better.

## precision recall f1-score support  
##   
## n 0.82 0.80 0.81 46  
## p 0.81 0.83 0.82 46  
##   
## accuracy 0.82 92  
## macro avg 0.82 0.82 0.82 92  
## weighted avg 0.82 0.82 0.82 92

**Classification Report for Sentiment**

The model is able to preform with an 82% accuracy meaning that it can correctly report sentiment in the general range of what would be expected for a model and could be used on new data to confidently help the restaurants identify scenario’s where a response and correction maybe needed.

As mentioned earlier with MultiNomialNB model is is easy to extract words with high performance. After doing a little bit of research on Stack Over Flow the code was found to extract the top and bottom n features from the model after training. For both the lie and sentiment models it was the same words

## -8.5241 $ 100 -7.5129 restaurant   
## -8.5241 $ 6 -7.5227 great   
## -8.5241 $ 7 -7.5593 best   
## -8.5241 % 's -7.5680 food   
## -8.5241 % service -7.5740 wa   
## -8.5241 & -7.8170 amazing   
## -8.5241 & pop -7.8205 good   
## -8.5241 & restaurant -7.8904 's   
## -8.5241 'after -7.8941 place   
## -8.5241 'after reading -7.9047 nice   
## -8.5241 'after went -7.9104 service   
## -8.5241 'bill -7.9179 fresh   
## -8.5241 'bill gate -7.9212 friendly   
## -8.5241 'brown -7.9241 friend   
## -8.5241 'brown tofu -7.9364 always   
## -8.5241 'd 'more -7.9390 price   
## -8.5241 'friday -7.9726 like   
## -8.5241 'friday worse -7.9751 need   
## -8.5241 'in diner -7.9848 quality   
## -8.5241 'just -7.9878 delicious   
##   
## C:\Users\kyles\ANACON~1\lib\site-packages\sklearn\utils\deprecation.py:103: FutureWarning: Attribute `coef\_` was deprecated in version 0.24 and will be removed in 1.1 (renaming of 0.26).  
## warnings.warn(msg, category=FutureWarning)

**Least and Most Informative Features**

As noted above the top 20 words include those usually associated with positive sentiment like great, best, amazing, good, fresh which is likely how the model was able to pick up on the majority of sentiment tags. Likely sarcasm not included. Given that the results match for both lie and sentiment is a bit surprising and clearly a different feature set would be needed for lie detection. Parts of speech tagging used in NLP might be a good example of a different trial if more time allowed.

# Conclution

As seen in this model, the Naive Bays preforms well when completing a sentiment task, however it does not support deployment with current techniques when applied to lie detection. This could be because the features that indicate lying were not readily part of the data set and would need other techniques or maybe a different model.

However; the high performance of the sentiment analysis could be provided to the restaurants for help in identifying when a customer has an issue. The Restaurant could then take potential action to remedy the given situation or look for a pattern of an issue among many reviews. This could server their bottom line well and quickly turn around a bad experience for their customers.