Sentiment Analysis of Yelp Reviews using NLP Techniques

Chenhao Shangguan, Kaiyu Zhang, Kun Bai, Yindi Ma

12/17/2021

Introduction

Customers' comment/review is a valuable resource to business owners, which can help owners better understand the quality of offering products and services. Yelp is a company that publishes crowd-sourced reviews about businesses and is commonly used by customers in the US. It also offers a reservation service for businesses, and customers generally check the comments before deciding to arrange reservations. Yelp users rate the service by a 5-star rating scale, and some of them would leave comments. However, the comments were not necessarily consistent with the rating. The inconsistency offers little insight to business owners. Thus, we started this project to address this problem and increase the accuracy of classification.

We use a sentimental analysis approach to achieve our goal. Sentiment analysis is a natural language processing (NLP) technique used to determine whether data is positive, negative, or neutral[1]. Sentiment analysis is a powerful tool that enables business owners to understand customers' emotions in their marketing campaigns. It is an essential and valuable tool for business owners who want to check product performance for strategy adjustment.

Our question of interest is whether using the NLP approach to implement sentiment analysis in large-scale data will provide an accurate prediction (0.7) or not.

The dataset we used containing 560,000 training samples and 38,000 testing samples of Yelp reviews. Each data entry has two columns, the first column is the attitude, based on the stars from the customers' review(1 = Negative, 2 = Positive). The second column is the text content of the reviews. This dataset comes from Kaggle.com and is extracted from the Yelp Dataset Challenge 2015 data. The files train.csv and test.csv contain all the training samples as comma-separated values. There are 2 columns in them, corresponding to class index (1 and 2) and review text. The review texts are escaped using double quotes ("), and any internal double quote is escaped by 2 double quotes (""). New lines are escaped by a backslash followed by an "n" character, that is "\n".

Method:

The language of Python and libraries of Numpy, Scipy, Pandas, and BER were used to conduct our analysis. The analysis was implemented with three main steps: Data Pre-processing, Word Embedding and Model fitting and prediction.

a) Data Pre-processing

In the first task, text preprocessing, we followed the preprocessing procedure consisting of uniformity, lemmatization, and removing rare words. In the uniformity part, the text's stop words were removed to recognize the lower and upper case words, which are the same. Additionally, stop words do not contribute to the subjective content for the sentiment of the comment. Common stop words are "a", "the", "is", "are", etc. After removing the stop words, all letters were modified into the form of the lower case. The next step is

lemmatization. In this part, we converted all the words into their root form to better categorize and reduce dimensions of data. For example, "words" can be transformed into "word". In this step, we utilized the WordNetLemmatizer from nltk.stem. Lastly, rate words with low-frequency appearing were removed since we do not want a model overfitted, and the rare words can contribute trivially to the model accuracy.

b) Word Embedding

The second part of our analysis is word embedding. By many literature reviews, the two main methods we used are Count Vectorize and TF-IDF.

Count Vectorize Count Vectorize is a relatively naive approach. Vectors that have the same dimensionality with the size of the vocabulary will be created and counted one and added to the previous value as one particular vocabulary was detected. A matrix containing vocabulary and counts will be generated by reading through each sentence. Each column is a particular vocabulary count vector, and each row is a sentence. This is a straightforward approach as the matrix illustrates the frequency of terms in each review, and some statistics such as mean, standard deviation, and variance can be easily computed and interpreted.

	terrible	excellent	lamb	expensive	date	place	vegetable	soup
terrible lamb, expensive	1	0	1	1	0	0	0	0
excellent date place	0	1	0	0	1	1	0	0
excellent vegetable soup	0	1	0	0	0	0	1	1

Table 1. Example of Count Vectorize matrix of three reviews after pre-processing

TF-IDF TF-IDF is a statistic that evaluates how important a word is to a document as an asset of documents. TF(t,d), which is known as term frequency, is defined as the number of times a word(i) appears in a document(j) divided by the total number of words in the document. A higher weight is given when the frequency of a particular word is high. TF is normalized by dividing the document length.

$$tf_{ij} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

IDF(t), which is known as inverse document frequency, is the log10 of the ratio of total number of documents to the number of documents with term t. It denotes the rarity of the word. If a word is very rare on occurrence, heavy significance would be addressed in IDF. TF-IDF is a statistic that is $TF(t,d) \times IDF(t)$.

$$IDF(t) = log(\frac{N}{df_t})$$

c) Classification

Then, we started to do the sorting work. We determine the tone of the comments by dichotomy. First, we divide the data into training and testing datasets. And then, we use the criteria from the training to generate a prediction and compute the accuracy.

Here we choose three efficient classifiers: Multinomial Naive Bayes Classifier, Support Vector Machine, Logistic Regression.

1. Multinomial Naive Bayes Classifier Here, we apply the Bayes theorem to conduct the classification and compute the probability of each vector in each class by prior information from the training set. And we put the specific vector into the class with a higher probability.

In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature, which is the meaning of "Naive".

According to the Bayes theorem and simple derivation, the likelihood of observing a histogram x is given by:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

$$P(c|x) = P(x_1|c)P(x_2|c)...P(x_n|c)P(c)$$

$$P(x|c_k) = \frac{(\sum_{i=1}^{n} x_i)!}{\prod_{i=1}^{n} x_i!} \prod_{i=1}^{n} P_{ki}^{x_i}$$

With this theoretical formula, we can conduct the classification

Step 1: Convert the data set into a frequency table, so that we could compute the class prior probability and predictor prior probability.

Step 2: Create Likelihood table by the probabilities calculated above.

Step 3: Use the Naive Bayesian equation to calculate the posterior probability for each class. As a result, the class with the highest posterior probability is the outcome of prediction.

2. Support Vector Machine Another machine learning method to classify the data is the Support Vector Machine method (SVMs). SVMs are supervised algorithms especially for binary classification tasks.

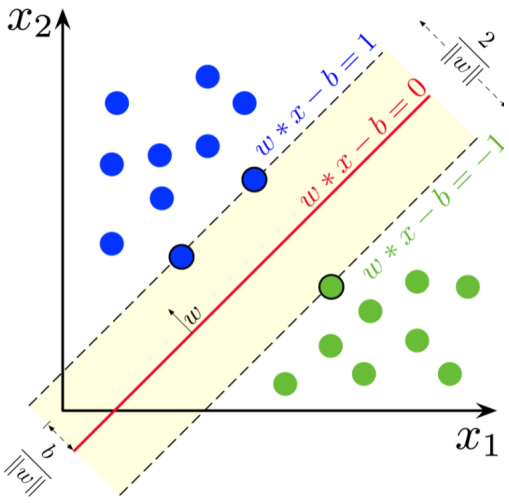

Figure 1: Theory of SVM

The figure above shows that support vectors is pointing in the data closest to the classification hyperplane. The hyperplane would be the red line constrained with the equation: $(w \times x - b = 0)$. It classifies the data by the signal of the vector w, which is the vector that begins perpendicularly from the hyperplane of the points. And the hyperplane would be the criteria of the classification. And our goal is to find the separating hyperplane that minimizes the loss by optimization and decide to classify.

Step 1: Minimize the loss function by the restricted formula and get the optimal hyperplane from the training dataset.

$$min(\lambda||w||^2 + [\frac{1}{n}\sum_{i=1}^{n} max(0, 1 - y_i(w^T x_i - b))])$$

Step 2: Then we can classify our data from the testing dataset by the signal function.

$$f(\overrightarrow{x}) = sign(\sum_{s} \theta_s(\overrightarrow{x}, \overrightarrow{x_s}) + b)$$

3. Logistic Regression Logistic regression is a statistical machine learning algorithm that classifies the data by considering outcome variables on extreme ends and tries to make a logarithmic line that distinguishes between them.

This model creates a regression model to predict the likelihood that a given data entry belongs to the category labeled "1."

Logistic regression fits the data using the sigmoid function, much as linear regression assumes that the data follow a linear distribution.

$$logit(E(Y_i|X_i)) = logit(\pi_i) = log(\frac{\pi_i}{1 - \pi_i}) = \beta X_i$$

Also, a threshold can be established to forecast which class a data belongs to. The derived estimated probability is categorized into classes based on this threshold. And our threshold would be 0.5.

Step 1: Fit the logistic regression by our training dataset and get the threshold as our classifying criteria.

Step 2: Test data by logistic regression. If the predicted value is less than 0.5, categorize the sentence into the negative group; otherwise, label it into the positive group.

Results

	Naive Bayes Classifier	Support Vector Machine	Logistic Regression
TF-IDF vectors	0.818	0.858	0.872
Count vectors	0.868	0.931	0.935

Table 2. The accuracy of the model classification.

From the results of Table 2, we found that classification accuracy is high since all the accuracy percentages are more significant than 80%.

In addition, we could find that our model prediction accuracy to classify count vectors is higher than that to classify the TF-IDF vectors.

For comparing classifiers, the accuracy of Logistic Regression would be the highest, which means that Logistic regression classifies the vectors most precisely with an accuracy rate of 87.2% for TF-IDF vectors and 93.5% for count vectors.

Discussion

After checking the accuracy, we found that the Naive Bayes Classifier has the least precise classification, whereas the Logistic Regression classifies the vectors the most precisely. That would be explained by the calculation rationales underlying the methods. For Bayes Classifier, we need to compute the probabilities; however, the logistic regression would fit the model, which would acquire more computation. According to that, it is a time-costing classifier than Bayes Classifier. That is a trade-off between speed and accuracy.

We have tried the Neural Network methods, but due to the trade-off effects, the speed of the Neural Network Classifier is too slow to get the results. The reason might be that the Neural Network is suitable for multi-level classification questions. When it comes to binary classification, the computation complexity decreases the effectiveness.

Therefore, in the future, we could add more classification levels to check the attitude of one sentence by other deep learning methods. Since the binary classification has only two levels, we cannot distinguish between the solid positive and neutral but slightly positive reviews, which is crucial for predicting one sentence's sentiment in NLP problems.

References

[1] Fang, X., Zhan, J. Sentiment analysis using product review data. Journal of Big Data 2, 5 (2015). https://doi.org/10.1186/s40537-015-0015-2