Sentiment Analysis on Twitter Data using KNN and SVM

Sentiment Analysis

Sentiment essentially relates to feelings; attitudes, emotions and opinions. Sentiment Analysis refers to the practice of applying Natural Language Processing and Text Analysis techniques to identify and extract subjective information from a piece of text. A person's opinion or feelings are for the most part subjective and not facts. Which means to accurately analyze an individual's opinion or mood from a piece of text can be extremely difficult. With Sentiment Analysis from a text analytics point of view, we are essentially looking to get an understanding of the attitude of a writer with respect to a topic in a piece of text and its polarity; whether it's positive, negative or neutral.

In recent years there has been a steady increase in interest from brands, companies and researchers in Sentiment Analysis and its application to business analytics. The business world today, as is the case in many data analytics streams, are looking for "business insight."

In relation to sentiment analysis, I am talking about insights into consumer behavior, what customers want, what are customers like and dislike about the products, what their buying signals are, what their decision process looks like etc because in the end of the its the customers for whose satisfaction these businesses work for.

```
#getting emotions using in-built function
mysentiment_covid19<-get_nrc_sentiment((covid19_tweets))

#calculationg total score for each sentiment
Sentimentscores_covid19<-data.frame(colSums(mysentiment_covid19[,]))

names(Sentimentscores_covid19)<-"Score"
Sentimentscores_covid19<-
cbind("sentiment"=rownames(Sentimentscores_covid19),Sentimentscores_covid19
)
rownames(Sentimentscores_covid19)<-NULL


#plotting the sentiments with scores
ggplot(data=Sentimentscores_covid19,aes(x=sentiment,y=Score))+geom_bar(aes(
fill=sentiment),stat = "identity")+
  theme(legend.position="none")+
  xlab("Sentiments")+ylab("scores")+ggtitle("Sentiments of people behind
the lockdown tweets on Covid19")
```

I tried using a variety of features for the classification experiments. For the baseline, I use word feature, n-gram feature, pattern feature and punctuation feature. Finally, include another key based feature. After calculating weight based on features for each tweet, I used these techniques.

In **Word Feature,** each word in tweet considered as a binary feature. For counting the word feature of a tweet, it compares with a dictionary which is used to detect which words are stop words and which are not. Otherwise, every word is considered for word feature. Moreover, if I encounter sequences of two or more punctuation symbols inside a tweet, I consider them as word features. Additionally, the common word RT, which means "retweet", does not constitute a feature because it may appear in the majority of tweets inside the dataset. When I calculate word feature weight we calculate is as (Nf/count f) where *Nf* represents the number of feature present in the tweet and count (f) represent the number of total feature present in the whole dataset. We use this formula for all the features of our experiment.

In **N-Gram Feature,** a sequence of 2-5 consecutive words in a sentence is regarded as a binary n-gram feature. The tweet which contains more rare words that have a higher weight than which contain common words and it has made a greater effect on the classification task.

In **Pattern Features,** the words are divided into three categories. They are high-frequency words (HFWs), content words (CWs) and regular words (RWs). When a word frequency is considered as *f* which frequency in the dataset is represent as *frf* and it will be considered as a high-frequency word if *frf > FH*. On the other hand, if *frf < FC*, then *f* is considered to be a content word and the rest of the words are considered as regular words. The word frequency is calculated from all the words of the dataset and it is estimated from the training set rather than from an external corpus. We treat as HFWs all consecutive sequences of punctuation characters as well as URL, REF, TAG and RT meta-words for pattern extraction as they play an important role in pattern detection. A pattern is an ordered sequence of HFWs and slots for content words. The upper bound for FC is set to 1000 words per million and the lower bound for FH is set to 10 words per million. FH is set to 100 words per million and we provide a smaller lower bound as the experimental evaluation produced better results. Observing that the FH and FC bounds allow overlap between some HFWs and CWs.

In **Punctuation Feature,** the last feature type is divided into five generic features as:
1) tweet length in words,
2) number of exclamation mark characters in the tweet,
3) number of question mark characters in the tweet,
4) number of quotes in the tweet, and
5) number of capital/capitalized words in the tweet.

The weight *wp* of a punctuation feature p is defined as wp = (3*Np)/ (Mp * (Mw + Mng + Mpa)) where *Mw*; *Mng*; *Mpa* declare the maximal values for word, n-gram and pattern feature groups, respectively. So, *wp* is normalized by averaging the maximal weights of the other feature types.

In the **Key-based feature,** I use the list which was previously gathered.

Experimental Result & Performance Evaluation

In this section, based on the result of Sentiment Classification Algorithm (SCA) and Support Vector Machine we try to evaluate the performance of different algorithms. Our key parameters to evaluate performance are Accuracy, Recall, and Precision, etc. We have used a few open source machine learning libraries during performance evaluation.