Assignment 4 – Text mining, Sentiment analyses
Due: Nov 21th , 2020

This assignment involves text mining of user-review data and sentiment analyses. It is based on a collection of reviews and accompanying star ratings from Yelp. A sample of the original dataset (over 4 million review by over a million users for 144K businesses) will be used here, to keep the assignment task manageable. We will examine the effectiveness of different sentiment 'dictionaries', and develop and evaluate classification models to help predict sentiment polarity (negative, positive).

The star ratings will be used here to indicate the sentiment label. For binary classification, we will need to convert the 1-5 scale rating values to {positive(1), negative(0)} values.

(More details on the data are available from https://www.yelp.com/dataset)

The data was given in json files. The reviews data file contains the reviews and includes reviewID, businessID, businessName, the review text, star rating and other attributes. The business data file contains the businessName, businessID, address, categories (restaurants, beauty and salon, food, fitness, local services, etc.), various attributes of the business (free wifi, wheelchair access, parking, smoking allowed, operating hours, … etc). Note that a business can fall under multiple categories, and these are specified in different variables names Category1, Category2,…

We will consider reviews for restaurants. The data has been pre-processed to get the business type, review text, star rating, and how many users found this review to be cool, funny, useful, into a single file which you will use for the analyses. There are ~ 44K rows in the sample file given.

Note: based on computing power available, you may need to take samples of the data to run different analyses in the questions below. Please indicate clearly if this was done, and what sample sizes you used (should be consistent for each team, i.e. different team members should not be working with different sample sizes). For building models, a minimum of 10K reviews should be considered.

We will use the "bag of word" approach for text mining, with standard steps for creating the document-term matrix (word vectors for each document; each row as document) - with either binary term presence/absence values, term occurrences, or tf-idf values. The steps are:

- Tokenize
- Transform case (to all lower/upper)
- Filter stopwords
- (optionally) Filter tokens by length - say, min 3 and max 15.
- (optionally) Filter tokens by content – if they match a 'dictionary' of terms
- (optionally) Stemming, Lemmatization
- others as you find useful.

You may also wish to deselect words that occur in too few documents and/or in most of the documents.

The optional steps above will be what you experiment with to determine what works best.

(a) Explore the data.
How are star ratings distributed? How will you use the star ratings to obtain a label indicating 'positive' or 'negative' – explain using the data, graphs, etc.?
Do star ratings have any relation to 'funny', 'cool', 'useful'? Is this what you expected?

(b) What are some words indicative of positive and negative sentiment? (One approach is to determine the average star rating for a word based on star ratings of documents where the word occurs). Do these 'positive' and 'negative' words make sense in the context of user reviews?
(For this, since we wish to get a general sense of positive/negative terms, you may like to consider a pruned set of terms -- say, those which occur in a certain minimum and maximum number of documents).

(c) We will consider three dictionaries, available through the tidytext package – the NRC dictionary of terms denoting different sentiments, the extended sentiment lexicon developed by Prof Bing Liu, and the AFINN dictionary which includes words commonly used in user-generated content in the web. The first provides lists of words denoting different sentiment (for eg., positive, negative, joy, fear, anticipation, …), the second specifies lists of positive and negative words, while the third gives a list of words with each word being associated with a positivity score from -5 to +5.

How many matching terms are there for each of the dictionaries?

Consider using the dictionary based positive and negative terms to predict sentiment (positive or negative based on star rating) of a movie. One approach for this is: using each dictionary, obtain an aggregated positiveScore and a negativeScore for each review; for the AFINN dictionary, an aggregate positivity score can be obtained for each review. Are you able to predict review sentiment based on these aggregated scores, and how do they perform? Does any dictionary perform better?

(d) Develop models to predict review sentiment.
For this, split the data randomly into training and test sets. To make run times manageable, you may take a smaller sample of reviews (minimum should be 10,000).

One may seek a model built using only the terms matching any or all of the sentiment dictionaries, or by using a broader list of terms (the idea here being, maybe words other than only the dictionary terms can be useful). You should develop at least three different types of models (Naïve Bayes, and at least two others of your choice ….Lasso logistic regression (why Lasso?), xgb, svm, random forest (ramger).

(i) Develop models using only the sentiment dictionary terms – try the three different dictionaries; how do the dictionaries compare in terms of predictive performance for rating ? Then with a combination of the three dictionaries, ie. combine all dictionary terms.
Do you use term frequency, tfidf, or other measures, and why? What is the size of the document-term matrix?
Should you use stemming or lemmatization when using the dictionaries?

(ii) Develop models using a broader list of terms (i.e. not restricted to the dictionary terms only) – how do you obtain these terms? Will you use stemming here?

Report on performance of the models. Compare performance with that in part (c) above.

How do you evaluate performance? Which performance measures do you use, why.