IDS 572 Fall '22

Assignment 3 – Analyzing text in Yelp reviews - Text mining, Sentiment analyses

Due: Nov 20th , 2022

This assignment involves the analyses of text data on users' review of restaurants and sentiment analyses. It is based on a collection of reviews and accompanying star ratings from Yelp. To keep the assignment task manageable, a sample of the original dataset (over 8 million reviews by over a million users for 160K+ businesses) will be used here. We will conduct an exploratory data analyses, examine the effectiveness of different sentiment 'dictionaries', and develop and evaluate predictive models to help identify 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 ratings to {positive(1), negative(0)} values.
(More details on the data are available from https://www.yelp.com/dataset)

The data was given in multiple json files (as detailed in the website mentioned above). 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 45K 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 words" 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.

1. Explore the data.
   (a) How does star ratings for reviews relate to the star-rating given in the dataset for businesses (attribute 'businessStars')? Can one be calculated from the other?

   (b) Here, we will focus on star ratings for reviews. How are star ratings distributed? How will you use the star ratings to obtain a label indicating 'positive' or 'negative' – explain using the data, summaries, graphs, etc.?

2. What are some words in the restaurant reviews indicative of positive and negative sentiment – identify at least 20 in each category.
   One approach for this is to determine the average star rating for a word based on star ratings of documents or reviews where the word occurs. Do these 'positive' and 'negative' words make sense in the context of user reviews for restaurants being considered?
   (For this, since we'd like 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).

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

   (a) How many matching terms (i.e. terms in your data which match the dictionary terms) are there for each of the dictionaries?
   (b) What is the overlap in matching terms between the different dictionaries? Based on this, do you think any of the three dictionaries will be better at picking up sentiment information from you text of reviews?
   (c) Consider the positive and negative terms you determined in Q 2 above; which of these terms match with terms in each of the three dictionaries?

4. Consider a basic approach (not developing a predictive model like a decision tree, random forests etc.) to use the dictionary based positive and negative terms to predict sentiment (positive or negative based on star rating) of a review. One approach for this is: based on 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.
   (a) Describe how you obtain the aggregated scores, and predictions based on these scores
   (b) What is the performance of this approach (for each dictionary). Does any dictionary perform better?

5. 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).

You should consider models built using only the terms matching the sentiment dictionaries, as well as 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, random forest (use ranger for faster run-times) – use the same three modeling techniques with each of the dictionaries, with the combination of dictionary terms, and wit the broader set of terms.

(a)How do you evaluate performance? Which performance measures do you use, why?

(b)Which types of models does your team choose to develop, and why?
Do you use term frequency, tfidf, or other measures, and why?

(c) Develop models using only the sentiment dictionary terms – try the three different dictionaries; how do the dictionaries compare in terms of predictive performance? Then with a combination of the three dictionaries, ie. combine all dictionary terms.
What is the size of the document-term matrix?
Should you use stemming or lemmatization when using the dictionaries? Why?

(d) 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 or lemmatization here, and why?

(e) Compare performance of the models. How does performance here relate to that from Question 4 above. Explain your findings (and is this what you expected).

6. Consider some of the attributes for restaurants – this is specified as a list of values for various attributes in the 'attributes' column. Extract different attributes (see note below).

(a) Consider a few interesting attributes and summarize how many restaurants there are by values of these attributes; examine if star ratings vary by these attributes.

(b) For one of your models (choose your 'best' model from above), does prediction accuracy vary by certain restaurant attributes? You do not need to look into all attributes; choose a few which you think may be interesting, and examine these.

Note: for question 6, you will consider the values in the 'attribute' column. This has values of multiple attributes, separated by a '|'. Further, some of the values, like Ambience, carry a list of True/False values (like, for example, Ambience: {'romantic': False, 'intimate': False, 'classy': False, 'hipster': False, …}. Care must be taken to extract values for different attributes. You can consider developing a separate dataframe with review_id, attribute, and then process this further to extract values for the different attributes.