Text mining And Sentiment Analysis

1) INTRODUCTION - WEB SCRAPING

Our initial purpose is to scrape relevant pieces of information about Amazon products, including product details, reviews, and ratings. We construct two scraping functions that take as input an URL and will return a tibble with the product information. The first function scrapes the product general info (product details, number of customer ratings, and the fastest delivery date), while the second one scrapes the reviews titles, texts, and stars. The product details are displayed on a single page, so our function needs to input only the product's ID. The reviews are organized on multiple pages, so our function will need not only the product ID but also the number and "position" of the pages from which we want to scrape. The reviews are also divided into "Reviews from the UK" and from "Other countries". In our particular case, we chose a book called " Fiesta: The Sun Also Rises (Arrow Classic)" by Ernest Hemingway.

2) THE DATA

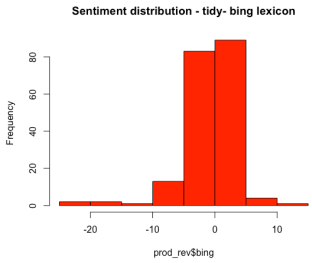
The initial data called product reviews contains 300 product reviews (comments) about the book, which are also characterized by an evaluation in terms of stars (from 1 to 5). To have a first look at the data we are working with, we extract a numeric score from the stars strings, and then plot it with a histogram of the review's stars. The histogram shows a high frequency of positive reviews among the overall comments, more than 90 reviews have five stars, and less than 30 reviews have only 1 star. We also compute a frequency table based on a binary classification of positive and negative reviews. We consider the reviews with less than four stars as negative and reviews with four stars or more as positive. From this classification, it appears that positive reviews represent 66% of the total and negative ones 34%. We can also use a box plot to compare the length of positive and negative reviews. In our case, positive reviews are on average shorter than negative ones and they also are characterized by a larger variability due to the presence of more outliers.

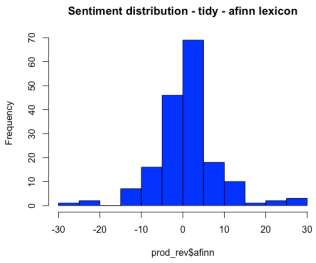
3) RESULTS AND DISCUSSION

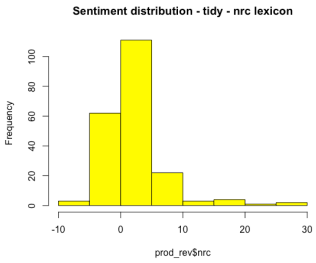
CLEANING AND PRE-PROCESSING. Once briefly analyzed the data we are working with, we have to proceed to the pre-processing and cleaning of the data itself. As we previously said, the product's reviews are not only from the UK, so the texts can be in languages different from English. With language detection we are able to filter only the English-written reviews, so we do not have to translate the non-English ones. In our particular case, we end up with 242 observations from the starting 300. Then we proceed with the tokenization of our data, which consists of extracting the words contained in the text of the reviews. Then, we have to clean our tokenized data. At this point, we want to eliminate stop-words (that are not relevant for our analysis). In our specific case, we build our custom stop-words (because we have a problem with the ' symbol that it is not detected when it appears as '). We also drop numbers (both integers and decimals) and a punctuation error that could cause us some problems (word. word are identified as a single word). We can now have a more precise idea of the data with a frequency table. The last part of our data cleaning process is word normalization, we can use stemming or lemmatization, which are two alternative methodologies. Stemming is the process of reducing the word to its root eliminating the suffix. Lemmatization is the process of determining the lemma of a word based on the context and identifying the part of speech of each word. For interpretability lemmatization is better but more computationally expensive. We implemented both as an example but we will proceed principally with the data from the Stemming process. All the choices made in this pre-processing and cleaning prices will have an impact on the output of the analysis.

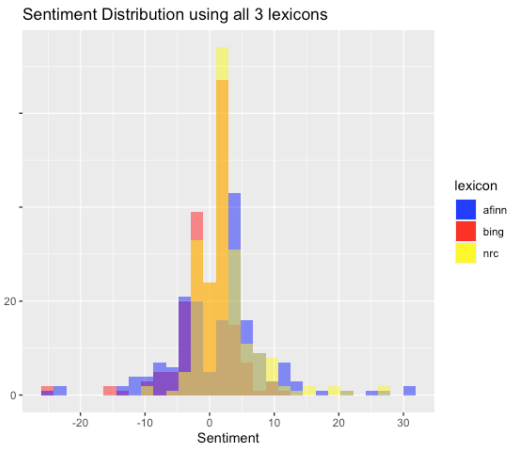
SENTIMENT ANALYSIS (DICTIONARY-BASED)

TIDYTEXT APPROACH. With this SA approach, we will use three lexicons: BING (gives words a positive or negative sentiment), AFINN (rates words with a value from -5 to +5), and NRC (labels words with six possible sentiments or emotions). As we previously said we consider words as tokens. The procedure for each of these lexicons is similar, but the results are dependent on the lexicon itself. With every specific lexicon, we are able to give a sentiment or value to (almost) each word, and then we compute the value of each



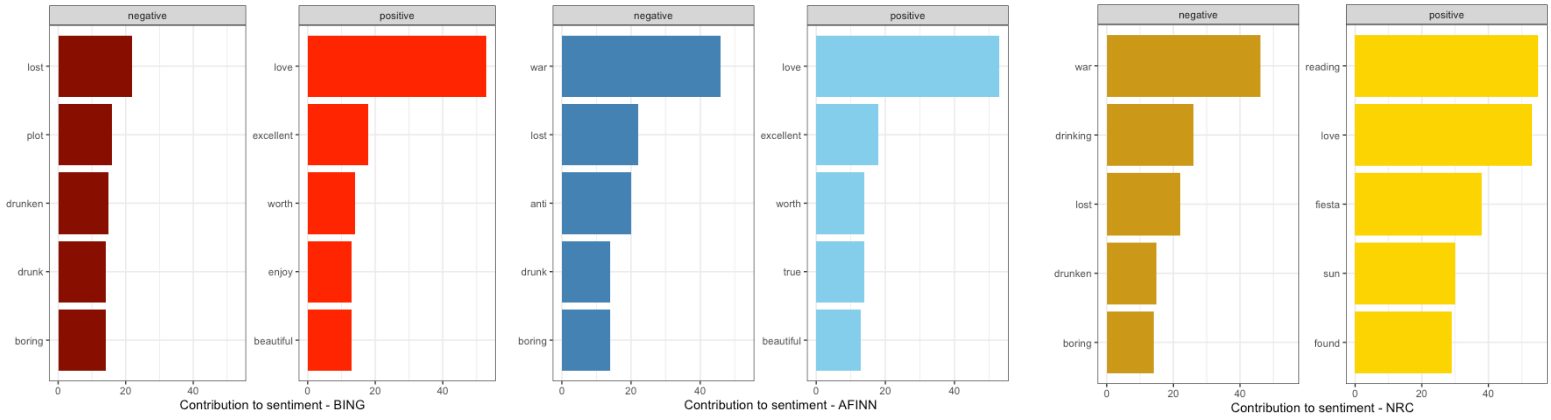






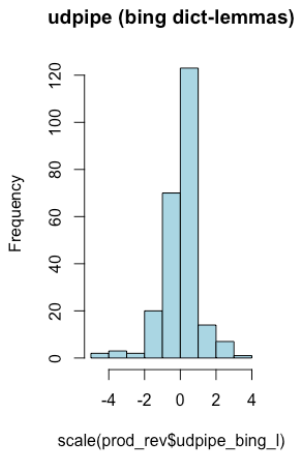
review as an aggregation of the contained words' values/ sentiment. We can plot our results using histograms.

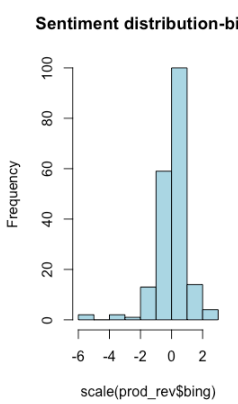
We plot the first histogram using the BING lexicon, the AFINN lexicon for the second one, and the NRC lexicon for the third one. In the last histogram, we plotted all three previous histograms to have an easier comparison of them. Between the three different evaluation, the NRC lexicon gives us a more positive view (mean = 2.452, median = 1.00), the AFINN one it is just a bit less "optimistic" (mean = 1.474, median = 2.00), while the BING lexicon it is on the negative side (mean = -0.4513, median = 0.00). We can also analyze the contribution of every word to the sentiment with both a bar chart and a word cloud. As we said the lexicons are built differently so, we try to represent a bar chart of the more significant words per sentiment (we consider only positive or negative sentiment for each lexicon, so we can have a better comparison).

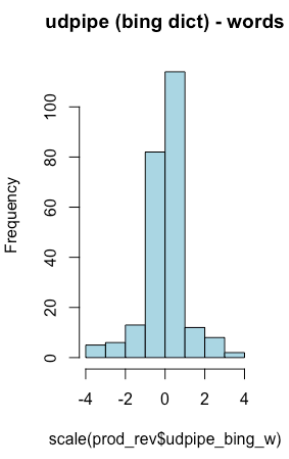


As we can see there are some similarities, but also some important differences (especially in the NRC-positive section) due to the different built of each lexicon. We can see similar results using the word-cloud representation.

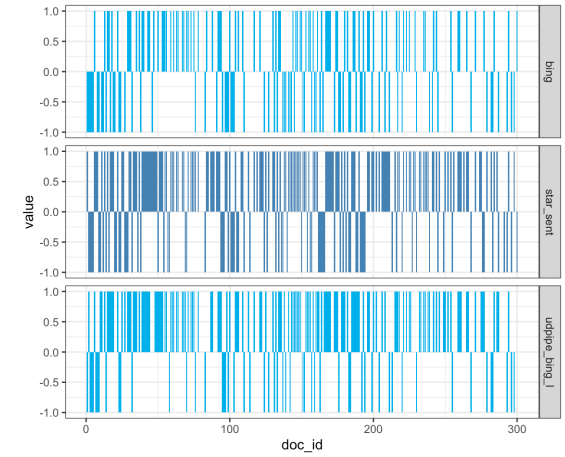
UDPIPE APPROACH. With this approach, we also consider polarity negators and polarity amplifiers. The performance increases when we consider them both. However, also this approach is not free from possible problems, there is some situation in which the approach under-perform the previous one. We can use lemmas or words in the analysis and we can use one from the three lexicons. We choose to represent only the BING case for both words and lemmas (the other two lexicons are used in the R code). We can see the histogram in the left and the one on the right are sentiment distribution (respectively considering lemmas and words), while the sentiment distribution in the center is the one relative to the tidy approach. The difference between the use of words and lemmas is in our case almost irrelevant. A relevant

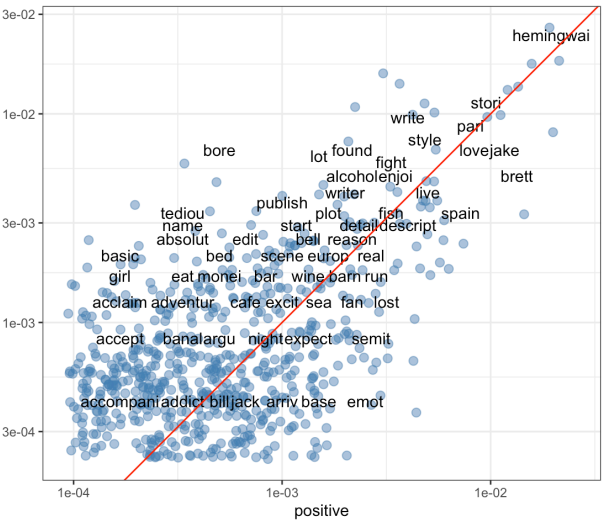


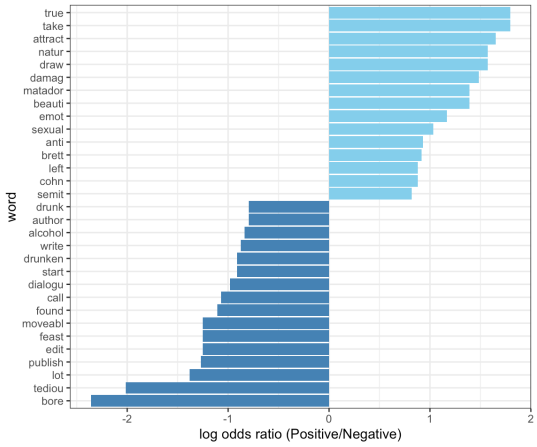




difference between the tidy approach and the udpipe approach using words is due to the elimination of stop-words in the tidy approach that is not dropped in the udpipe one; however, the different results between the two approaches are principally caused by the different approaches themselves. For the BING case, the udpipe approach brings us a distribution more concentrated on the positive values than the tidy approach (mean = 0.7669 instead of mean = -0.4513). Also for both the AFINN and NRC lexicon, the udpipe approach brings sentiment distribution relevantly more concentrated on the positive sentiment and with a larger mean. We can now build a really basic and naive analysis to investigate if the sentiment reflects the reviews' stars. It is clear that there are some differences. We can also compare the sentiments with the star score (pretending that it is the true one). These results strongly depend on the pre-processing phase. For the tidy approach, as said before, we eliminated strop-words. For the udpipe one, we considered lemmas instead of words (as said do not generate a big difference) and we didn't remove stop-words.







VISUALIZATIONS

We can provide different visualization for a better understanding of the data. UNIGRAMS. Unigrams consist of a single item from a sequence. We start by looking at the most frequent stems in all our documents. Then we can compare the stems by people that wrote a positive review and the ones from negative reviews. To show that, we can use different types of visualizations. We can start with two simple bar plots for positive and negative reviews. We notice that the stems used are pretty similar. So, in order to show which stems are important but specific to each category, we can compare the frequency of stems in positive and negative comments. The stems which lie near the red line are used with about the same frequency in the two categories. We can also compare the log-odds ratio to understand which words are likely to come from each category of reviews. A different type of visualization could also be a word-cloud representation.BIGRAMS. We can replicate some of the previous plots also for bigrams. In our code, we consider a new type of visualization, that allows us to understand better the relationship between words. More precisely, if you are interested in the relationship between words, with also the possibility of representing the link between them or the concurrency of words (words following each other).

4) CONCLUSION

After having scraped product information and reviews we cleaned the data. We were also able to see how the choices that we made in the pre-processing and cleaning part can influence the output of the entire analysis. The choice of dropping stop-words and the choice of stemming for word normalization have brought us to different results than if we chose the opposite alternatives. We also saw the differences between a dictionary-based SA using the tidy approach and using the udpipe one. Also, in the tidy approach, we saw how the choice of different lexicons can give us different results, due to how those lexicons are constructed. We then have covered some of the various visualizations that we can use to describe our data, using unigrams, bigrams, and others. The reviews are mostly positive, and that we knew from the start, but it is important to see how different choices and approaches can bring us to different results, and how such an analysis can be done from multiple and different prospectives.