Project 2 proposal: Sentiment Analysis on Movie Reviews

By Noel Mathew

What is the problem I want to solve?

Every time a movie comes out, there is always somebody reviewing it. Sometimes the movie could get great reviews. In other cases, the movie could be a flop and get bad reviews. This project goal is to evaluate what type of model can predict a good review or a bad review the best. In addition, this project will check if any of these words can show if a review is good or bad. This means does certain words show that the movie is a classic or a flop.

Who is my client and why do they care about this problem?

One of the clients in this case will be the movie critics. The movie critics can use this project to learn which phrase of words strongly highlight if a movie is bad or good. Furthermore, they can use the project to figure out what phrase to use to get to the point of their review. Another client could be the general audience who can use the solution of the problem to write user reviews on sites like Rotten Tomatoes. Another client could be people who work in the movie industry. For example, a movie producer or executive like Disney and can run the model on multiple reviews online to see which movie was good or bad and decide which are good movies to make sequels from. This will help companies save time and resources in manually looking at each review and determining that the movie is good or bad

What data am I using? How will I acquire the data?

The data I will be using is a movie phrase train set which has phases from reviews and their sentiment analysis score. I will use this training set to help fit a model to the test set. I will use the test-set to assign a sentiment label score to the phrase. All the data will come from https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews/data and https://www.kaggle.com/c/word2vec-nlp-tutorial/data.

Briefly outline how I'll solve this problem

I believe that various machine learning methods will be used to see relationships between phrases and words and their sentiment towards a movie. I will run many different performance scores to see which model can show the relationship between words and sentinement the best.

Here's my outline as I envision the project right now:

1. Gather, wrangle, and tidy up data for movies reviews I can get data for
2. Train various natural language processing models I can on the data,
3. Use cross-validation techniques to tune hyper-parameters for each of the model
4. Use the results of the model to see what model was best in predicting if the movie was good or bad.

What are my deliverables?

At the end of the project, I will have a Jupyter Notebook with both the project code and my report comments. I will also have a slide deck for presenting the project. I will have a list of strong sentiment phrases both positive and negative why phrases have such a strong sentiment towards a movie.

Data Cleaning

I got the data to do the sentintement analysis for movie reviews from https://www.kaggle.com/c/word2vec-nlp-tutorial/overview/part-2-word-vectors. The data set was used part of a Kaggle Word2Vec and natural language processing tutorial and competition. The file I wanted to use to do the modeling was called labeledTrainData.tsv. The file contains 25000 reviews from IMDB. It contained three fields, the id for the movie, the review of the movie, and the sentiment score. The sentiment score is 0 for a negative review and 1 for a positive review. Therefore, the problem I want to solve is a binary classification one.   The file was converted to a dataframe using pandas package and its DataFrame method. The reviews did not only have words in them. It had html tags like <br> and "/". They also had punctuation which needed to be removed as it will mess up building the vectors and thus mess up the machine learning. There was numbers in reviews which could have the same effect as punctuation. The non-words had to be removed from the review and the next paragraphs will explain that process.

In order to get rid of the html tags, the beautiful soup library was used. The Beautiful soup library is a library that is used for pulling data out of HTML and XML files. This library is needed as the reviews are html encoded or have html tags. So I got each raw review from the dataframe and got rid of the HTML using BeautifulSoup().get_text() method. Then I got rid of the numbers and punctuation by using regex expression to replace nonwords with spaces. Then the words in the review was converted to lowercase because Uppercase and lowercase of the same word has different values. So if one word has both an uppercase and lowercase form then those would different values even though they are the same word. Therefore, by converting the words to lowercase it keeps consistent values among the words. The review was split into a list with each item being a words, so the review can be iterated through. Then set of stopwords was brought in. A stopword is a word that has no or neutral meaning. For example, the word “the” is a stopword because it provides no meaning on context to the content. This in turn can make a review more neutral than it is. So, a set was made with those stopwords like ‘the’. Then the stopwords was removed in the list of words in the review by making a list comprehension that excluded the stopwords. Then new list of words was join back to one string. This was done in one function called review_to_words.

The review_to_words function was run on each review on the dataset. Then each review got appended to a list called clean_train_reviews. Then a new column in dataframe called cleanReview was made and the values were the items in clean_train_reviews. Then dataframe was converted to a csv file using pandas; therefore the csv file had the clean reviews in it.

Exploratory Analysis

The new csv file with the clean review was brought over using the dataframe feature from pandas. The dataframe had 25000 entries and 4 columns. The 4 columns are ‘id', 'sentiment', 'review', 'cleanReview'. The first three columns were like before and ‘cleanReview’ column is the new columns with reviews with just words. There are no null values in the datalist, so did not have to worry about imputing null values. The clean reviews are all strings, so data cleaning process was done correctly. Then prebuild word2vec model gensim was brought in to find words with similar meanings in reviews and see if those reviews containing those words were positive to negative. A word2vec model translates words into vectors that are plotted in a coordinate space. Therefore, one could find similar words by distance of the vectors between each word. So, see how

gensim finds similar words, it was tested on some words. For example, it was tested on the words sad. By using most_similar positive method of the gensim model it was able to find similar words to sad which were ('saddening', 0.7273085713386536), ('Sad', 0.6610826253890991), ('saddened', 0.6604382991790771), ('heartbreaking', 0.6573507785797119), ('disheartening', 0.6507317423820496), ('saddens_me', 0.6407118439674377), ('distressing', 0.6399092674255371), ('saddest', 0.6345508694648743), ('unfortunate', 0.6272086501121521), ('sorry', 0.6194046139717102). This shows the words that are closest to the word sad and their vector scores. Likewise this test was run on other words like good, bad, horrible, and fantastic. In order to see the distribution of the review scores, a histogram was used on sentiment score to see the frequency of positive and negative reviews. The distribution was even between positive and negative reviews which 12500 each. This could have been done on purpose as this data set came from a tutorial. They probably made the data set even to run accuracy score to check the performance of the tutorial models as it quick and easy performance test. The next analysis that I wanted to do was to see how a word in a review affects if a review as positive or negative. Therefore, I used the most similar method from the gensim model again. So, I started with the word fantastic because the word is very positive in nature. So, I got the similar words to fantastic which were 'terrific', 'wonderful', 'great', 'amazing', 'marvelous', 'fabulous', 'awesome', 'phenomenal', 'incredible', 'unbelievable'. Then a loop was ran and the reviews which contained those words were appended to the list. The list had a length of 9968 scores and those scores were converted to a dataframe. Then the scores were graphed using a histogram plot. The distribution was the reviews were mostly positive with some negatives. The negatives could come from the word like ‘unbelievable' For example a review could have been written as ‘the movie was ‘unbelievable' in its premise’. Therefore, that is a negative review. This was done on the word horrible as well because horrible is a negative word.
The most similar words to horrible were 'terrible', 'horrendous', 'dreadful', 'horrid', 'awful', 'atrocious', 'horrific', 'horrible_horrible', 'hideous', 'appalling’. The numbers of the reviews that contained the list of words above were 4618. The distribution of the score was mostly negative. The positive scores could come from reviews like “The dreadful nature of the villain was greatly fleshed out”. The next word that was tried was bad because it should be a common word used in negative reviews. The words similar to bad were 'good', 'terrible', 'horrible', 'Bad', 'lousy', 'crummy', 'horrid', 'awful', 'dreadful', 'horrendous'. It is weird to see good there, but this might mean that good is words that does not have a lot of positivity. The number of reviews that the words above were 14333 reviews and the distribution of the sentiment was mostly negative, but there is a good number of positive reviews. This could be that the word good appears in more reviews compared to the word bad as good is the only positive word in the list. The next word that was tried

was good because it should be a common word used in positive reviews. The words similar to good were 'great', 'bad', 'terrific', 'decent', 'nice', 'excellent', 'fantastic', 'better', 'solid', 'lousy''. It is weird to see bad and lousy there, but this might mean that good is words that does not have a lot of positivity or bad does not have a lot of negativity. Furthermore, good could be used negative manner like the phrase it's just good.   The number of reviews that the words above were 18757 reviews and the distribution of the sentiment was even. This could be that the word good has a double meaning and the word bad appears more often than the other words in the list. I did the same for best and worst and found their distributions to be even respectively. This could be due to the nature of the review. This could be done with more words, phrases, and combinations. but it will take a lot of time doing this for 100 words

Since these are unique reviews with words or text, it is not possible to do statistical analysis on the features which are the clean reviews. The labels have a mean of 0.5 as the scores are evenly disburtubed between negative (0) and positive (1).