SENTIMENTAL ANALYSIS ON AMAZON FOOD REVIEWS

Prof. Nik Bear Brown, Ajinkya Kunjir, Kruti Lakhani, Vedant Singhvi

Master of Science in Information Systems

Northeastern University, Boston, MA

Abstract- Sentimental analysis is the measurement of positive and negative language. It is a way to evaluate written or spoken language to determine if the expression is favorable, unfavorable, or neutral, and to what degree. In this paper we will discuss whether the reviews given by the people are positive or negative based on the ratings given. Reviews can even be classified on the basis of words used in comments as well. It helps us to answer many questions like 'Is the product good enough?', 'Is the given review helpful?', 'What are alternatives of the specific product?' and many more. We tackled this problem by using classification algorithm like logistic regression and Naïve Bayes algorithm. Using wordcount vectorizer along with nltk library and wordcloud helped to give more accurate results. Additionally, we have investigated the problem of ever-growing amount of text data available online and developed a text summarization model which shall generate short, accurate, and fluent summary of a longer text document. This was done with the help of deep learning algorithms like Bi-directional RNN, LSTM and seq2seq model.

Keywords: Sentimental Analysis, Natural Language Processing, Text Summarization, Wordcloud, Count Vectorization, ConceptNet, Recurrent Neural Network, Long Short Term Memory, Sequence to Sequence

I. INTRODUCTION

What makes a product good? What are the major concerns of the customers regarding specific products? Common reviews may give the general feedback like good taste, bad smell, not so good taste and many more. All these reviews are just the general reviews and we cannot predict the actual rating of the product using these reviews as there is lot of hidden information in the reviews [1]. Consider an example, a customer may write positive review about a product but rate the product as neutral. It can confuse other people by looking at the review and rating simultaneously. Hence in this paper we are going to overcome these hurdles and correctly predict the product reviews and ratings as positive or negative using sentimental analysis [2]. Sentiment Analysis is the process of determining whether a text data is positive or negative depending on the occurrence of the words. It is widely used in applications such as automatically detecting feedback towards products, news and characters or improving customers' relation model. Sentimental uses the concepts of Word Embeddings, Bag-Of-Words model and classification algorithms to classify text according to sentiment [3].

The ever-growing text data, in the form of email, social media posts and blogs has paved the way for summarizing data and generating summaries of long textual data. Various complicated algorithms have been developed to counter this problem, and they have seen relative success. We have deployed Seq2Seq

algorithm developed by Google, which incorporates Recurrent Neural Network with Bi-Directional Neural Network.

II. METHODS

A. Methods/ Algorithms used

1. Logistic Regression
2. Word embeddings
3. Recurrent Neural Network (RNN)
4. Long Short Term Memory (LSTM)
5. Sequence to Sequence Model

1. Logistic Regression

Logistic Regression is a regression analysis to conduct when the dependent variable is binary. Like all regression analyses, the logistic regression is a predictive analysis as well. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables [4]. Naive Bayes classifier is also another classification and predictive model. It is most widely used and gives more accurate predictions than any other prediction algorithms.

Instead of running individual codes to check the accuracy of the model, we will create a ROC Curve. The curve with the highest AUC value will show our "best" algorithm. ROC stands for Receiver operating characteristic. We use the matplotlib library to plot the ROC Curve.

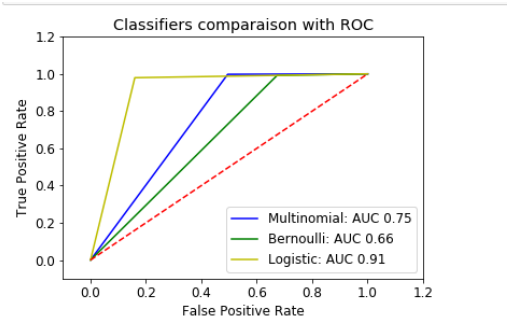

Figure 1: Classifiers comparison with ROC

Hence, we can figure out that Logistic Regression shows more better performance than Naïve Bayes Classifier for our data.

2. Word Embeddings

Word Embeddings are the texts converted into numbers and there may be different numerical representations of the same text. It tries to map a word using a dictionary to a vector. It is primarily of 2 types:

1. Frequency based Embedding
2. Prediction based Embedding

In our project, we have TF-IDF and Count Vectorizer for Sentiment Analysis of Text data, and Pre-Trained word vectors for the Auto-Summarization [6]

- Count Vectorizer: The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. The CountVectorizer indexes terms by descending order of term frequencies in the entire corpus, not the term frequencies in the document. After the indexing process, the term frequencies are calculated by documents.
- TF-IDF: It stands for Term Frequency-Inverse Document Frequency. It is based on the frequency method but it is different to the count vectorization in the sense that it takes into account not just the occurrence of a word in a single document but in the entire corpus [7].
- Pre-Trained Word Vectors: These set of vectors are large-scale data, preprocessed and learnt over a long time. In our project, we have used the Concept Net Numberbatch set of word vectors.

3.Recurrent Neural Network (RNN)

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. In a traditional neural network we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If you want to predict the next word in a sentence you better know which words came before it [9]. Thus, where RNNs come in, where the they perform the same task for every element of a sequence, with the output being depended on the previous computations.

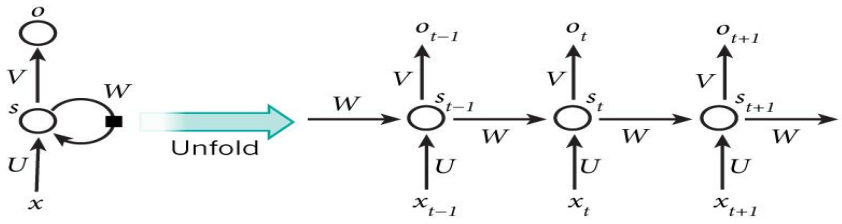

Figure 2.3: RNN network layer

4.Long Short Term Memory

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). A RNN composed of LSTM units is often called an LSTM network. The expression *long short-term* refers to the fact that LSTM is a model for the short-term memory which can last for a *long* period of time. An LSTM is well-suited to classify, process and predict time series given time lags of unknown size and duration between important events [8]. LSTM consists of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three *gates* can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as *regulators* of the flow of values that goes through the connections of the LSTM; hence the denotation "gate".

5.*Sequence to Sequence Model*

Seq2seq is a general-purpose encoder-decoder framework for Tensorflow that can be used for Machine Translation, Text Summarization, Conversational Modeling, Image Captioning, and more. The Sequence to Sequence model (seq2seq) consists of two RNNs - an encoder and a decoder. The encoder reads the input sequence, word by word and emits a context (a function of final hidden state of encoder), which would ideally capture the essence (semantic summary) of the input sequence [10]. Based on this context, the decoder generates the output sequence, one word at a time while looking at the context and the previous word during each timestep.

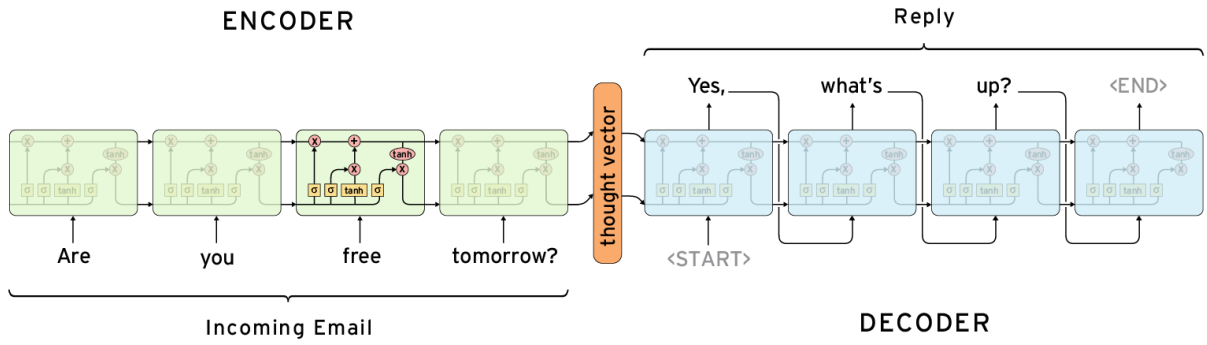

Figure 2.5: Sequence to Sequence model

III. RESULTS

To classify words as good or bad, we used wordcloud which is a graphical representation of words. It represents words using different criteria, shapes and colors. The following figure shows the generated positive and negative words based on NLTK library.

Figure 3.1: Positive words using wordcloud

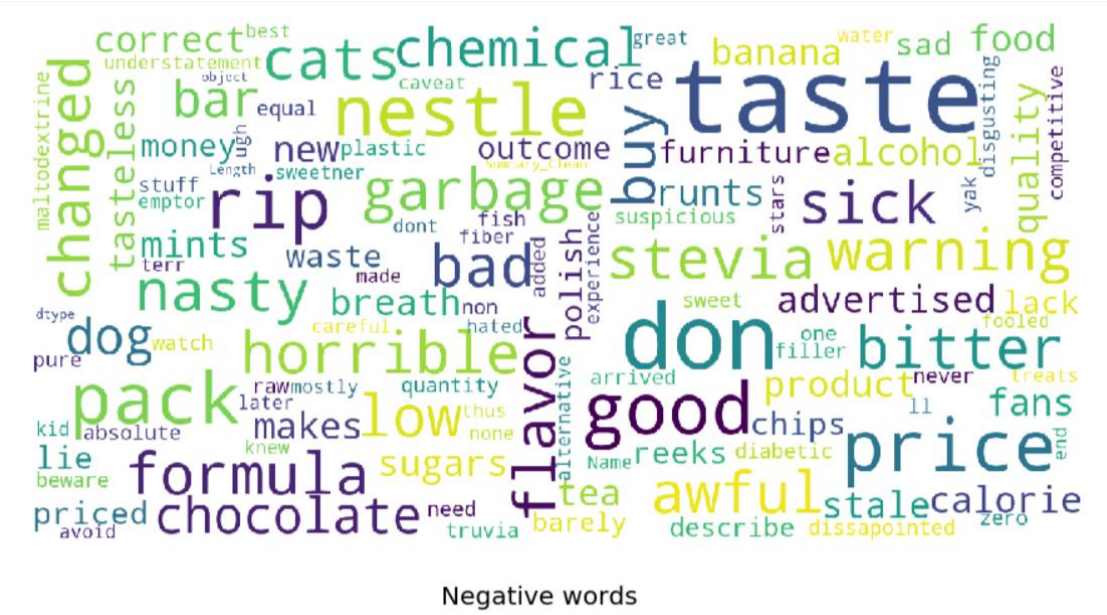

Figure 3.2: Negative words using wordcloud

As we implemented the TF-IDF along with n-grams we got a better prediction of positive and negative words according to the coefficients assigned. The accuracy of the model is almost about 94%. The following figure shows the positive and negative words along with their coefficients:

516481	loves	17.996218
285459	finally	18.302457
412619	heaven	18.743899
278773	favorite	19.179195
508754	love	19.445658
597872	not too	19.480049
348185	good	19.957199
212068	delicious	20.983266
17504	amazing	21.144665
262545	excellent	21.469968
598519	not very salty	21.818800
592568	not horrible	22.647332
961167	wonderful	23.058999
274671	fantastic	23.154943
596053	not so bad	23.247915
59276	awesome	23.448799
655587	perfect	23.547822
589080	not bitter	26.726139
588631	not bad	34.122757
368375	great	35.500419
81124	best	42.347969

Figure 3.3: Weights assigned to words using Count vector and TF-IDF

Based on worcloud and TF-IDF plus n-grams along with NLTK library we predicted the review as positive/negative. Below figure shows the output generated classifying the reviews.

```
#Lets predict whether the text samples are positive or negative

def test_sample(model, sample):
    sample_counts = count_vect.transform([sample])
    sample_tfidf = tfidf_transformer.transform(sample_counts)
    result = model.predict(sample_tfidf)[0]
    prob = model.predict_proba(sample_tfidf)[0]
    print("Sample estimated as %s: negative prob %f, positive prob %f" % (result.upper(), prob[0], prob[1]))

test_sample(logreg, "I just started using this product so I think it is too soon to rate")
test_sample(logreg, "Food was delicious but smell was awful")
test_sample(logreg, "The food was ok, I guess. The smell wasn't very good, but the taste was ok.")
```

```
Sample estimated as POSITIVE: negative prob 0.001472, positive prob 0.998528
Sample estimated as NEGATIVE: negative prob 0.998375, positive prob 0.001625
Sample estimated as NEGATIVE: negative prob 0.543000, positive prob 0.457000
```

Figure 3.4: Estimating samples as positive/negative

Using Deep Learning methods like RNN and bi-directional LSTM along with seq2seq vector we predicted short summary for long reviews which will help customers to get an overview of the review by just merely reading the summary instead of whole review. Following figure shows the generated summary for randomly generated review form the dataset.

```
INFO:tensorflow:Restoring parameters from E:/Studies/ADS/best_model.ckpt

 REVIEW:
 surprised intensity chocolate flavor cookie <br >it slightly drier nana cookies tried probably loaded cocoa powder

 Word Ids:    [17267, 52006, 35239, 19681, 39116, 37444, 42860, 5063, 45159, 28191, 51419, 9147, 37126, 37830, 40897, 20619]

 SUMMARY
  Response Words: chocolate heaven
  Word Ids:       [35239, 4327]
```

Figure 3.5: RNN and LSTM output predicting short summary for a given long review

IV. DISCUSSIONS

The dataset consists of half a million Amazon fine food reviews. This dataset is incredibly simple in structure, but that simple structure belies its richness. It includes a complex graph of product-reviewer relation as well as rich text data. We have come up with interesting avenues of exploration like what words tend to indicate positive and negative reviews? What makes a review helpful? Which product is most favorable? How does the review score distribution vary across reviewers?

By applying predictive algorithms like logistic regression to classify between positive, neutral and negative reviews, we got 94% accuracy which clearly shows that the method applied is correct and logical. We have created interesting bar graphs to predict User behavior. We have also applied Deep learning techniques- LSTM and RNN on our dataset to obtain a generalized title for the review given by a customer. The sequence-to-sequence model uses bi-directional LSTM, which helped in connecting previous information to the present task, thereby, generating more accurate results than a normal LSTM model would generate.

V. REFERENCES

[1] https://arxiv.org/ftp/arxiv/papers/1709/1709.08698.pdf

[2] https://cs224d.stanford.edu/reports/WuJi.pdf

[3] https://ieeexplore.ieee.org/document/8022740/

[4] https://www.statisticssolutions.com/what-is-logistic-regression/

[5] K. Sparck Jones. "A statistical interpretation of term specificity and its application in retrieval". Journal of Documentation, 28 (1). 1972.

[6] https://en.wikipedia.org/wiki/N-gram

[7] http://ceur-ws.org/Vol-1866/paper_72.pdf

[8] https://sertiscorp.com/thai-word-segmentation-with-bi-directional_rnn/

[9] http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

[10] https://www.tensorflow.org/versions/r1.0/tutorials/seq2seq

VI. GITHUB LINK FOR PORTFOLIO

https://github.com/AjinkyaKunjir/Advance-Data-Science/blob/master/Sentimental%20Analysis%20on%20Amazon%20Food%20Reviews/Amazon_Portfolio.ipynb