Capstone Project: Seq2Vec Sentiment Modeling of App Review Data*

Chike Odenigbo[a,1,]

[a] *Vector Institute, 661 University Ave Suite 710, Toronto, ON M5G 1M1*

Abstract

Proactively tracking and improving sentiment for products has become a source of competitive advantage for many businesses, who now have access to large crowdsourced datasets. Blogs, social media and news articles all serve as treasure mines for businesses to better gage the reception of their products by their consumer base and society at large. For my capstone project, I will be going over the findings of my analysis of HuggingFace's App Review Dataset [1].

Contents

*This document is the capstone project for Vector Institute's Intensive Introduction to Machine Learning Course.

1. Introduction

The goal of this project is to build a machine learning model that is able to predict the number of stars on an app review given the text input of the review. In this case, the stars that the user gives, which ranges from 1 (negative) to 5 (positive), is considered as a sentiment score that a supervised machine learning model can learn from. Many models were analyzed for this paper with mean absolute errors ranging between 0.65 and 1 meaning that the score given on a review would on average be expected to be off by less than 1 star.

1.1. Applications in Industry

In the world of data science, structured data is the tip of the iceberg when it comes to machine learning applications. Unstructured data, which is data that is unlabeled, accounts for about 90% of all data [2]. In other words, the abiltiy to capture instrinsic information about unstructured data is a great opportunity for firms to see the bigger picture and make smarter decisions. The models that were built in this project can be made to predict sentiment on text data sources that are similar in vocabulary and review length to the text data on which the model was trained. If there is a data drift whereby the reviews start naturally changing which causes a decrease in performance, the model would need retraining. As such, a production pipeline would be needed to report on the performance of the model on new reviews in the application store.

The value proposition of tracking the sentiment of app reviews is that businesses can better understand customer preferences. From giving businesses the power to better measure the performance of certain features in their apps through A/B testing to allowing businesses to proactively perform competitor analysis, businesses will be much smarter in their interactions with customers. Some points of consideration that businesses would have to pay attention to in production would be bot comments, sarcasm and what I call AI warfare (fake news) that can be used to throw off the model.

1.2. Exploratory Analysis

By looking at the distribution of the ratings in Figure 1, we can observe that there is a large number of very positive reviews. As a matter of fact, there are more reviews that have 5 stars than all other categories combined. In terms of modeling, the minority classes would potentially need to be upsampled in order to prevent the model from overfitting to the top class.

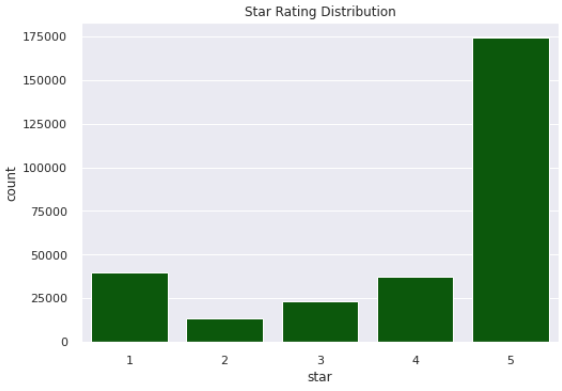

Figure 1: Ratings Distribution

In terms of the actual text in the review, Chi-Square tests were used to assess the importance of the words in relation to the number of stars in the review. The idea behind Chi-Square tests is that if a word (feature) is uncommon in general, but is common with a certain star rating (target), that word will have a higher Chi-Square score because it has a strong predictive power. As demonstrated in Figure 2, it is possible to generally derive the sentiment of the sentence given the words with the highest scores.

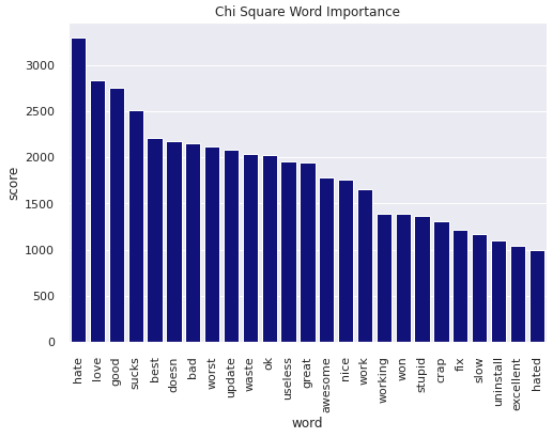

Figure 2: Word Importance

Analysis of further depth could have allowed for more insights. For instance, n-gram's could have be used to capture the importance of multiple words on the target. Latent Dirichlet Allocation (LDA) could have also been explored to determine topics in the data.

2. Modeling

2.1. Methodology

Given the ordinal nature of the target variable, regression techniques were used in the model building. Depending on the model, the words were converted into their numerical vector representations by using either a document-term matrix or word embeddings. The following are the models that were explored:

1. XGBoost Regressor with bag of words inputs (Baseline)
2. Deep Learning (RNN, LSTM, GRU & Regular Dense Networks) with pretrained word embedding layer from Google News' Swivel 20 dimensional embedding (Challenger)

Techniques such as balancing the target variable, adding bidirectionality and regularization of the models were also explored to optimize the performance. The expectation is that the recurrent network frameworks will perform better than the tree model because of their ability to process data that has a logical sequence (in this case sentences) as opposed to a bag of words model that processes the presence of words. Furthermore, the embeddings allow for better generalizability because they are better able to capture multiple words of similar meaning as opposed to the bag of words which again just captures presence of words.

2.2. XGBoost Regressor

2.2.1. Description

The document-term matrix that was used for XGBoost was the term frequency–inverse document frequency (tfidf) data structure. A tfidf matrix is an effective way of summarizing a series of documents (reviews) by providing a normalized count of each word in the entire corpus (dataset) by document. This normalized count of each feature (word) is then fed to the model which traverses the different combinations of word counts in the review and outputs its expected outcome. XGboost was used because of its ability to capture the presence of multiple interactions in the data. Its challenge in comparison to a recurrent network is its ability to capture logical sequences.

2.2.2. Performance

The XGBoost Regressor was able to achieve a mean absolute error of 0.75, however, this number could be misleading given how unbalanced the dataset is. It could just be predicting a majority of the reviews as having 5 stars which would generally be correct given the unbalance. As displayed in Figure 3, the model seems to have fit better on reviews with 4 or 5 stars than the rest of the classes. Solutions to this challenge could be to convert the model to a binary classification task of positive and negative reviews or to upsample the lower classes. In this paper, the latter was explored.

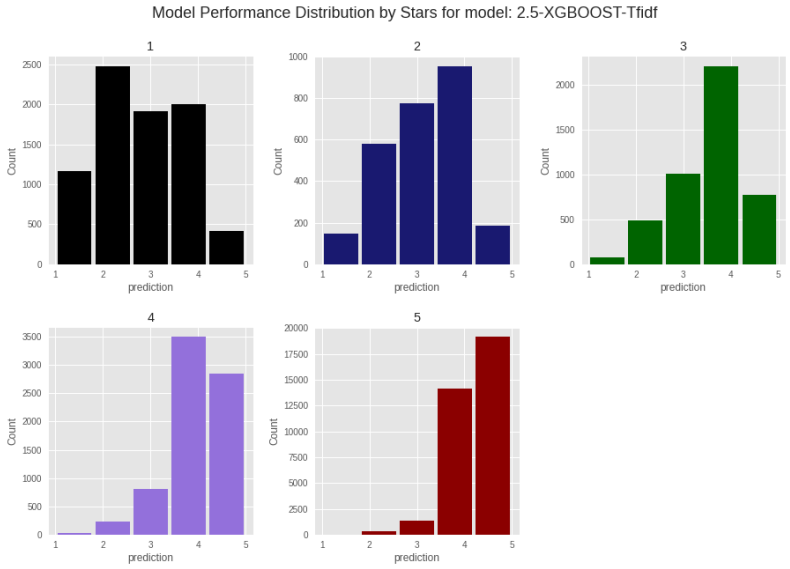

Figure 3: Model Prediction Distribution by Actual Value

Balancing the dataset on the other hand seems to have pushed the model towards predicting most values towards the middle. The model's mean absolute error was increased to 1.06, however it predicts far better the negative reviews.

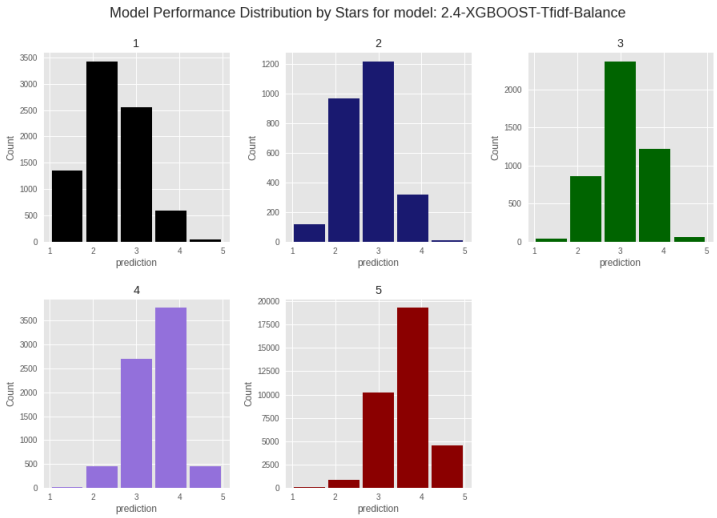

Figure 4: Balanced Model Prediction Distribution by Actual Value

2.3. Deep Learning

2.3.1. Description

Like the XGBoost model, deep learning architectures were explored because of their ability to capture interactions in the data, meaning that the model can capture combinations of different words with regards to the target. They are preferred to the tree models firstly because the embeddings capture more relational information than the count provided by the tfidf matrix. Word embeddings capture semantic meaning of words and represent them as numbers. As such one could make an equation of the vector representation of the words "Man", "King" and "Woman", and perform the equation "Man" + "Woman" - "King", and receive the vector representation of "Queen" or a word similar to "Queen" such as "Princess". The Swivel embeddings used in this project is derived from Google News and represents words in a 20 dimensional vector space [3].

The recurrent architectures (RNN, LSTM, and GRU) were explored because they are able to keep a state of information at each time step of the previous words in a sentence during the model training. The challenge with recurrent networks however is that gradients might vanish or explode particularly in longer sentences. GRUs and LSTMs were used to contain this issue because they can forget words that are unimportant to the sentence. Attention layers could have also helped to negate this issue.

2.3.2. Performance

In terms of performance, Table 1 shows that the LSTM and GRU models had the best results as expected with the LSTM being the champion model. Leveraging embeddings that were trained on a similar corpus or that have a higher dimensionality could have potentially helped to improve the model performance in this regard as well.

Model	Mean Absolute Error
Bidirectional LSTM with Dropout	0.664155659
Bidirectional GRU with Dropout	0.677971985
Bidirectional RNN with Dropout	0.690937809
Regular Dense Network with Dropout	0.695537466
LSTM without Dropout	0.706941142

Table 1: Top 5 Models

Another interesting finding is that all the models that were balanced performed much worse than the models that were unbalanced. My hypothesis for this was that the reviews with low scores could be ambiguous and contain many positive words. Upon manual review of the test cases that had errors, however, it is possible to notice that erroneous predictions would stem from:

- typos
- false negatives/positives from the user
- sarcasm
- contrarian topics in a given review
- punctuation
- random words such as names and slang

There would also be natural errors such as the model predicting a value that could arguably be correct. Table 2 is a random sample of the champion model's errors that were manually reviewed. One might argue that embeddings trained on a corpus of a similar domain would have performed better given that the Swivel news article embeddings used for the model tend not to have slang, sarcasm or the reflection of sentiment through punctuation. In short, this is a direct reflection of working with a real world messy dataset and brings back the point mentioned at the start of the paper on data drift and the need to monitor and retrain the model.

review	star rating	prediction	absolute error
J Devare	3	4	1
Damm ???????	2	4	2
It is brillant	5	4	1
Need Indonesia language...	5	3	2
Tt G	5	3	2
Good so far	4	5	1
Ya man its spr	3	5	2
It's good	4	5	1
salma	1	4	3

Table 2: Random Sample of Erroneous Predictions for Bidirectional LSTM with Dropout

In terms of distribution, the LSTM network seems to have fairer distributions between the positive and negative classes than all of the other models that were explored as displayed in Figure 5.

As such, the model would need to be further assessed and applied to predict on similar unstructured data such as Tweets for the apps in the dataset.

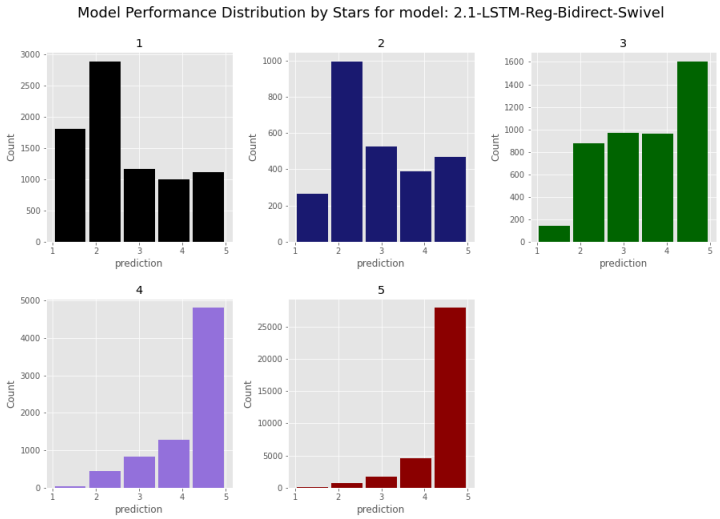

Figure 5: LSTM Model Prediction Distribution by Actual Value

Further considerations and fine-tuning might also be required to allow the model to adapt to the new data source. The model might however not be adequate in assessing large reviews such as blogs or articles as displayed with the higher mean absolute error in longer reviews in Figure 6. Depending on the use case, companies might also want a model that focuses on a particular area. For example a company might want a model with a higher recall on negative sentiments so as to focus on improving the concerns raised in those reviews.

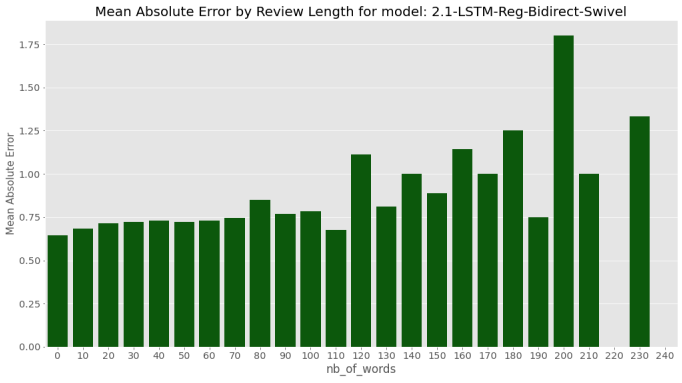

Figure 6: Mean Absolute Error by Review Length

3. Conclusion

In conclusion, text data is a prevalent commodity that is a big opportunity for many companies. To summarize the process, as long as words can be translated to numbers, value can always be extracted from the text.

References

[1] https://huggingface.co/datasets/app_reviews

[2] https://www.forbes.com/sites/bernardmarr/2019/10/16/what-is-unstructured-data-and-why-is-it-so-important-to-businesses-an-easy-explanation-for-anyone/?sh=7a131dfd15f6

[3] https://tfhub.dev/google/tf2-preview/gnews-swivel-20dim-with-oov/1