| Problem Chosen | 2020 MCM/ICM Summary Sheet | Team Control Number |
|---|---|---|
| C | | 2018901 |

# A Multi-source product evaluation model: actual combat with BERT-guided Deep learning and Statistical analysis


**Summary**

Online marketing has always been a place of fierce competition, and thus business companies are always forced to make careful evaluations before launching products. However, the current de facto evaluation and prediction models employed by ecommerce systems are largely based on the simple star ratings. These methods are irrational because they ignore the key information of the text reviews, and also fail to represent more recent opinions, rendering them unreliable in making time-based future predictions.

Herein, to overcome the aforementioned disadvantages of existing evaluation and prediction models, we proposed a multi-source model to evaluate a product. In our model, text reviews are first transformed to number by subjecting the 'washed' review text (with NLP techniques) to the pre-trained BERT transformer developed by Google, which performs word embedding with high efficiency, and the corresponding output was entered into multilayer Recurrent Neural Network (RNN) trained with IMDP movie review dataset, which attains training accuracy of 91.24% after three epoches. The obtained score for text review and the star rating were first weighted summed for each feedback weighted by helpfulness vote, after which the total product reputation is calculated as the weighted meridian, while the weight is determined by a linear decay algorithm. With the time-based decaying weighting method, opinions from new comments can be reasonably reflected, and the general performance of product over long time is less likely to be affected by false reviews.

Using our model, meaningful quantitative and qualitative patterns, relationships, measures, and parameters are identified within the provided datasets. We found that people tend to vote more frequently on the informative long reviews from the extreme groups which are generally regarded as more useful. The slight difference for microwave oven is well explained by its non-consumable product attributes. The weighted average calculated using our model integrates the product reputation and sale volume, two key factors in determining whether a product is successful. Using our model, we further found that specific star ratings do not usually incite certain type of reviews, indicating that customers are not easily influenced, possibility because that customers can obtain information on the product from multiple sources thus eliminating the effect of recurrent star ratings. Finally, using word cloud, we found that generally speaking, quality descriptors do associate with ratings, with positive ones appearing more in high ratings and negative ones in low ratings. However, some positive descriptors are also found within the low rating comments, and this may result from use of ironic expressions by reviewers.

Based on our model and the analysis results, we come up with strategies for Sunshine company in featuring their products. Using our model, the company can track the reputation and success of the product, as well as get insight into the future market.



**Keywords**: Natural Language Processing (NLP), Bidirectional Encoder Representation for Transformers (BERT), Recurrent Neural Network (RNN), time-related linear decay, Data Visualization.

# Contents

# 1 Introduction

## 1.1 Background

In the ecommerce market, every transaction has detailed records and user feedbacks, which could be exploited to provide guidance for future marketing. And the most frequently mentioned parameters in sales record including: (star rating), the gross experience of customers over the products; (text-based review), a short paragraph depicting the customer's personal experience with the purchases; and (helpfulness vote), other customer's opinion regarding the information conveyed by a certain review. In fact, the industry has been working on evaluating product reputation and making accurate predictions of the future market based on these data. However, the current widely used evaluation and prediction models employed by ecommerce systems such as Amazon, Cnet, Ebay and extra are based largely on the relatively simple star ratings, which only aggregate the numeric ratings using basic algorithms. On the one hand, this mode completely ignores the key information hidden behind the text reviews, especially at times when discrepancies between the actual meaning behind words and star ratings exist. On the other hand, the simple operation of evaluating a product with just a click on the mouse makes the results from model based on numeric star ratings easy to be manipulated. Moreover, although customers ideas towards the features of certain products changes with time passing by, the majority of existing models assign same weights to both the old and new reviews, therefore failing to reflect more recent opinions, further resulting in their unreliable performance in making time-based future predictions. Under this circumstance, building reasonable model to perform product evaluation and market prediction can help both the customer and manufactures to make meaningful product decisions.

## 1.2 Problem Restatement

Problem restatement To better understand the sales of a product, it's necessary to encompass the information of text reviews in our evaluation model. Based on the given data, we performed the following steps:

1. We performed data wrangling, during which all the words in text are reduced to its root forms to decrease the complexity of semantics.

2. Build a our emotional sentiment grader model: the pre-trained BERT was used to execute word embedding, and the output of BERT was entered into RNN. Train the grader model till the accuracy reached above 90%. Use this model to score all the text reviews.

3. Using statistical analysis to unravel relationships within and between star ratings, reviews and helpfulness votes. Use the parameters to test reliability of our grader model.

4. Based on the scores for reviews obtained using the above model, weigh sum the star rating and review score using helpfulness vote for each feedback.

5. Calculate time-related linear decay weight for each review, and calculate the weighted meridian as indicator of the product reputation, calculate the weighted average as indicator of product success.

6. Time-Series analysis to determine if certain kind of reviews will be incited by specific star ratings

7. Word map to unravel the correlation between rating and descriptors

8. Proposing sales strategy and product design feature based on the obtained results to the Market Manager of Sunshine Company.

9. Analyze the strength and weakness of our modelling.

### 1.3 Work and Goals

Our overall goal is to establish a model to accurately and intuitively evaluate a product and predict the market using multi-source information including star rating, text review, helpfulness vote, date, sales volume and etc. Validate the reliability of our model, use our model to reveal the relationship between parameters in sale record. Moreover, guided by the obtained results using our model, we hope to come up with online sales strategy and identify product design features for Sunshine Company to succeed in their field.

## 2 Model

### 2.1 Assumption

1. We assume that the majority of reviews offered by customers are honest to reflect real personal experience with the purchases, and there's no strong competitive relationship between customers and retailors, such that false reviews (eg: to damage a competitor's product reputation for personal interest) do not exist.

2. The data set provided by the Sunshine company comes from reliable source that truly reflects the selling data of the product under consideration.

3. Competition is not significant for different items (product ID) from the same product.

4. Each reviews are written by customer who has bought and used the product by person (no matter if they get the refund later).

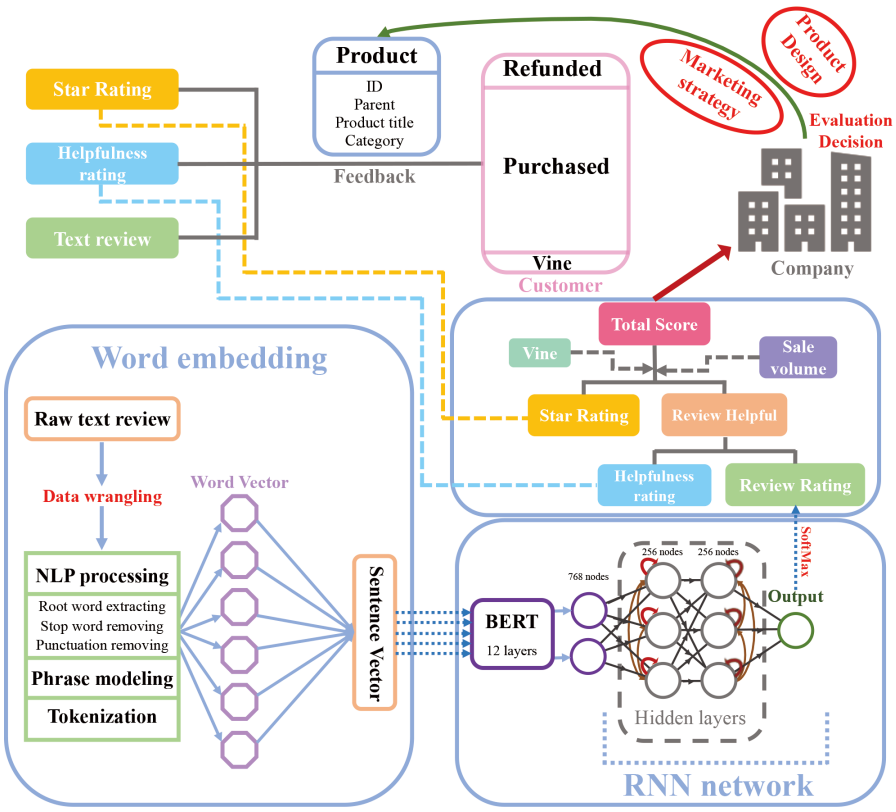


Figure 1: Illustration of our model (Details was discuss below)

## 2.2 Natural Language Processing: Sentiment Analysis

### 2.2.1 Word Embedding

Besides performing regular data washing procedures to deal with cases, punctuations, special line breakers, and etc. on the raw review text, we further employed the WordNetLemmatizer from the nltk library to reduce each word to its root form, and also used normalize from unicodedata toolkit to remove accents (such that naïve is undisguisable from naïve), as well as exploited the stopwords from nltk.corpus package to remove meaningless stop words such as pronouns, articles and predispositions. Furthermore, bigrams were modelled based on the frequency of simultaneous appearance of two adjacent words such that some essential meanings would not be lost.

## 2.3 BERT-based Deep learning

The key point of this problem is teaching computer to "understand" the reviews. This is a open frontier question called Nature Language Processing(NLP). The most successful and practicable method based on machine learning: The machine itself "finds" the feature of language from enormous amount of data. The key procedure of supervised learning which we use in our model is that the machine compares it's prediction with label data and reduce the difference by self-adaptive parameters adjustment.

In this article, we employed the Google's pre-trained BERT[2] (Bidirectional Encoder Representations from Transformers) model were use as our embedding layers which was fed in to a GRU to produce a prediction of the sentiment of the input text.

The obtained tensor from BERT would be the input of our subsequent RNN Network. The two modules are trained with the IMDB dataset with some of BERT parameters frozen during training to enhance prediction accuracy and efficiency, and the output data (array format) from were map to a value between 0 and 1. The nearer to 1, the more confience we have to say a review has positive sentiment. The network is illustrated in Figure1 at right-bottom corner. The dataset provide over 17K labeled reviews of IMDB users with positive or negative sentiment.

We refer to this open source Github project[1] when we finish the machine learning code. In our model, we use 17500 data to train, and 7500 are reserved for validation test. And 25000 are used to test the accuracy. And it reachs 91.24% accuracy(Figure 2) after 3 epochs of training.

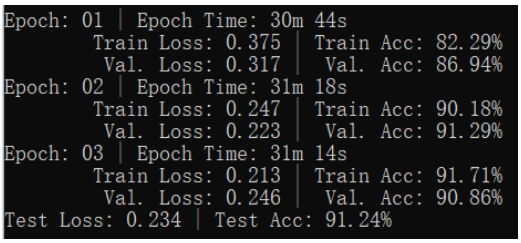


Figure 2: The accuracy of machine learning

When apply to our question, the validity of there sentiment scores are further tested.

## 2.4 Score

As mentioned before, the result of Sentiment Analysis is a number in [0, 1]. However, the user of Amazon may read any of these reviews and express their opinion by voting. In our model, we consider these votes as a weight of review rating which may finally influent the total score. How to deal with this votes is a question. Although more helpful votes imply that a review is more credible which should have higher weight when we count it into the evaluation, they may have considerable unhelpful votes which weaken the credibility. Both helpfulness votes and helpfulness rating should be considered. Which review is more reliable review, an 1-1 review or 439-480 one?

In our model, we think the unhelpful vote maybe more sensitive. The criterion of a review may suggest they haves some disadvantage or untruth. Considering all these factors, we balance the helpful and unhelpful, advice to use helpful rating vote as the estimation. Details was shown in Sec. 5

## 2.5 How Company use our result?

We get a rating for the each comment. To represent the "reputation" of product over time. Using the reputation, the company may track their customers reaction to their production. When reputation drop, there might be some risks of shrink of sales and potential damages to interests. So, the reputation score should be sensitive to time.

Further, we provide the world cloud which tells company how to improve their production. For example, if "easy" appears frequently, the guests may care about convenience and for "big", the guests may care more about appearance.

Both these data can help the company make decision and introduce new product to market. Further, the reaction of market will be collected and processed for further improve-

ment.

# 3 Results

## 3.1 Information Mining on star ratings, reviews and helpfulness ratings

### 3.1.1 Univariate analysis

We first dived into the star ratings, text reviews and helpfulness rating information of each product selling dataset and employed some conventional statistical analysis methods to identify patterns that could be instructive to retailors. As shown in Figure 3, we first grouped all feedbacks of each product by the star ratings gave by the reviewer. Although the sample size differs in each product, the results universally reflect that the majority of feedbacks remain largely positive (four stars and above), followed by negative (one and two stars) and neutral (three stars).

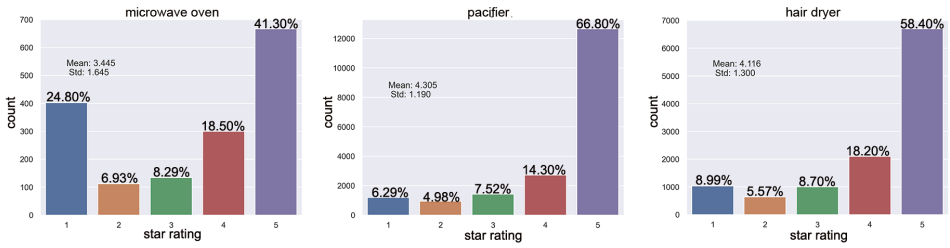


Figure 3: Number of feedbacks grouped by star rating for microwave oven(left), pacifier(middle) and hair dryer(right). The proportion of each group to the whole is indicated as percentages. Mean and standard deviation were labeled in each histogram.

Meanwhile, based on experience, the amount of information a text review could convey is related to its length, which could be represented by the number of words a piece of review encompasses. Thus, we plotted the probability density distribution map of word number in the text reviews for all three products (Figure 3), which exhibited a unimodal distribution with the majority of text comments consisting of 10 to 100 words. And it should be noted that small portion of reviews are "uniword", and we speculate that the involved word may be some recapitulative emotional adjectives.

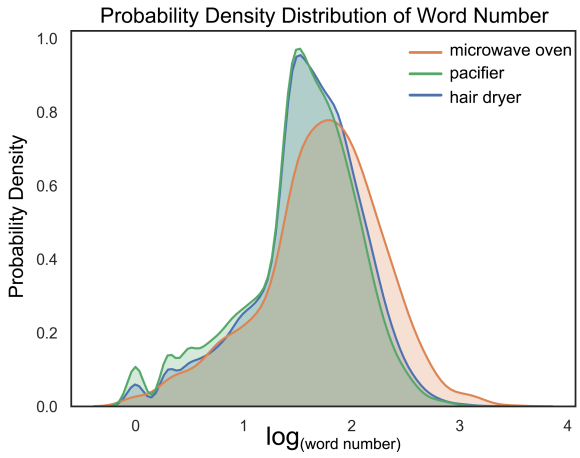


Figure 4: Probability density distribution map of word number in reviews from microwave oven (orange), pacifier (green) and hair dryer (blue)

Moreover, for the helpfulness ratings (Figure 4): The vast majority of feedbacks remained 'un-voted', while very few of the comments got high votes. This is in agreement with the "Exposure Effect" come up by Gustav Fechner and the common display mode by webpages whose item sorting is related to the number of views and votes, such that the already exposed items get more visits and thus more votes.

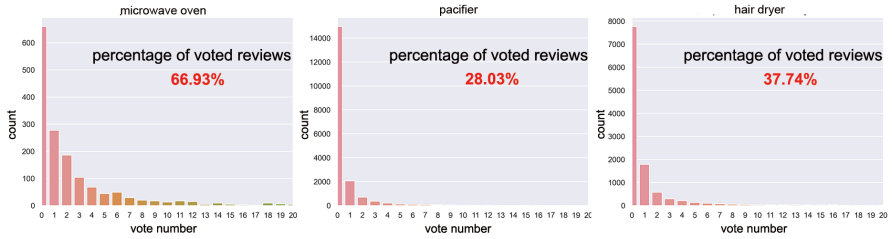


Figure 5: Number of feedbacks grouped by number of total votes regarding helpfulness of microwave oven(left), pacifier(middle) and hair dryer(right).

However, through a parallel comparison between products, we found that the distribution pattern differs slightly for microwave oven: a considerably larger portion of negative star rating was presented, the major distribution interval for review text length is greater, as well as significantly greater percentage of voted reviews (66.93%) and more feedbacks with a medium number of total votes regarding helpfulness (6-14) were found. After consideration, we assume that this circumstance may arise from the distinct commodity property of microwave oven, pacifier and hair dryer. Microwave oven is undoubtedly an example of expensive non-consumables, whose long-term quality and user experience matter a lot, under which condition purchasers are more likely to go through more feedbacks from previous customers before making a final decision, which explains well the presence of a greater proportion of voted reviews compared to the other items. In the context of high expectation towards products, customers have greater tendency to offer incisive reviews on their purchases, and are also more likely to comment negatively when the products fail to meet their expectations.

### 3.1.2 Bivariate analysis

**Star Ratings and Review** For ecommerce websites like Amazon, the star rating (an definite integer between 1 and 5) left by buyers is an important shorthand, symbolizing information about personal feeling towards the product. Although its high conciseness may deprive its ability to present detailed customer comments, the star ratings should represents similar distribution pattern as the that of the scores for text reviews obtained from our emotional sentiment model. By this notion, an applicable review emotional sentiment grader should be featured with the ability to distinct different feelings of text review from each star class in all three products. To test the reliability of our emotional sentiment model. We utilized the box plot to visualize our model's accuracy in grading review from each star class. As shown in Figure 5, the value obtained using our model separates well between different star groups, and the boundaries between positive (four stars and above), neutral (three stars) and negative (one and two stars) groups are very clear. One thing to be noted is that a minority of low star rating reviews received relatively high scores when analyzing the text review. This phenomenon may be caused by the imprecision of our model in complex semantic analysis (eg: irony words and phrases ), which leads to incorrect understanding and abnormal scores. But in general, our model is credible in scoring the text reviews objectively.

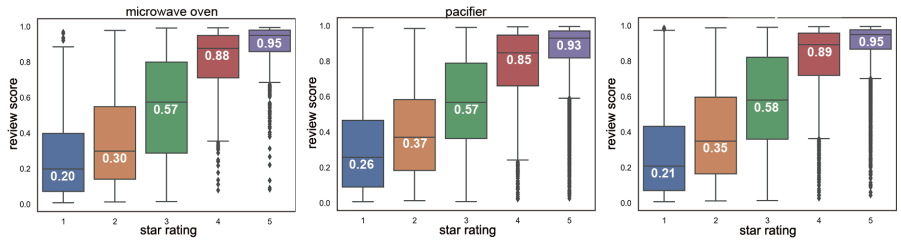


Figure 6: Box plot of review score grouped by star ratings for microwave oven(left), pacifier(middle) and hair dryer(right). The short dash on both sides indicate the edge of reliable data. The rectangle box indicated the range between the upper and lower quartiles. The line within the rectangle indicated the median, and the corresponding value is labelled below. Outliers are denoted as dots.

**Review and Helpfulness** In the meantime, the scatter plot of total number of helpfulness vote and the word number in text review (Figure 6) suggests that more votes are tend to be given to the long reviews, which is minor but is believed to contain more accurate and reliable information. Notably, the verified vines provided relatively long review, which are believed to be informative and are generally paid more attention to by customers.

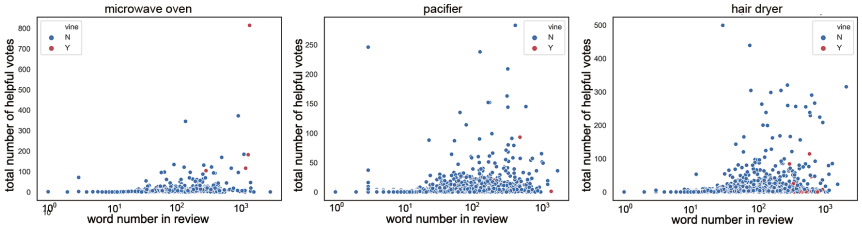


Figure 7: Scatter plot of total number of helpfulness vote and the word number in text review for microwave oven(left), pacifier(middle) and hair dryer(right). The verified vines are represented with red dots.

**Star Ratings and Votes on Helpfulness** To inquired into the simple relationship between star ratings and votes on helpfulness, we calculated the total of helpful votes in each star rating group (Figure 7), and the obtained result suggests that more votes on helpfulness distributed around the extremes of good (five star) and bad (one star) in all three products. This is logical since purchasers from the two extreme groups generally carries most strong feelings about the purchases, thus are tend to write text reviews that are relatively longer, which allows them to convey more information to the readers.

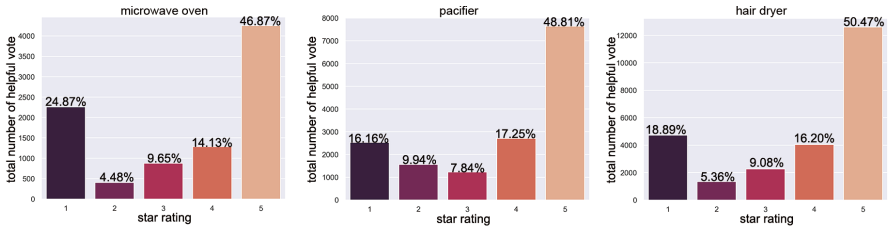


Figure 8: Total number helpfulness voting grouped by star rating in microwave oven(left), pacifier(middle) and hair dryer(right). The percentage indicates the proportion of votes on helpfulness from each group in total votes.

# 4 Sunshine Company Marketing Director's question

## 4.1 Total Scores

In order to measure the overall result of each review, we consider the results of both star ratings and reviews. By sentiment analysis as we mentioned, the full review text will be transformed into a score which described the emotional. The score is a number ranged from 0 to 1, where 1 means completely positive while 0 means completely negative. Then, in order to make the evaluation system consistent, we project the score for review to [-1,1] and project the star rating to [-1,1] as well.

As star rating is a more subjective and direct measurement for both the reviewer and other readers while the body part of the review is relatively more objective and indirect, we can regard the reviews are an supplement and adjustment to the star rating, so we place more weights on star rating instead of the review score shown as following.

$$F = 0.7 * star + 0.3 * H * reviewscore \tag{1}$$

H is the factor for helpful rate. It measures the trustworthiness of each review as higher h indicate this review is probably more credible. So, we multiple this factor to the review score which decrease the influence of those untrustworthy reviews. It is defined by the number of helpful votes and the number of total votes as following.

$$H = \begin{cases} helpful/total & total \neq 0 \\ 0.9 & total = 0 \end{cases} \tag{2}$$

H= number of helpful votes / number of total vote (if number of total vote!=0) 0.9 (if number of total vote=0)

## 4.2 Product's reputation

In this section we want to define a time-based measures which suggest the a product's reputation. In [3, 4], the writer suggested that median value is better choices than the mean. However, their criteria include informativeness, i.e. how likely is it that the ranking that a user finds at the time of making a choice will still be the ranking when the user is using the product or service[4]. This criteria is fundamentally conflict with our destination: we want a measure which fluctuate with time and suggest the increase and decrease of reputation. From this point of will, we suggest these criteria:

1. **Robustness**: The reputation is not distorted by outliers.
2. **Long Time Stability**: After saled for a long period of time, the reputation should be stead.
3. **Short time Fluctuations**: In a short period of time, some accidental events(scandals) may influence the reputation of a product, although they do not change the long time pattern. This may reflect the potential damage to company profits.

It seems time-weighted mean is the best choice. First: when number of review is large enough, the law of large number and center limitation theory guarantee the robustness and long time stability. Meanwhile, set a higher weight to recent data may cause short time fluctuations of products. We use the linear decay method raised in [3]

$$w_i = w_1 + (i-1)\frac{w_1}{n} \tag{3}$$

$w_1$ can be determined by normalization:

$$\sum_i^n w_i = 1$$

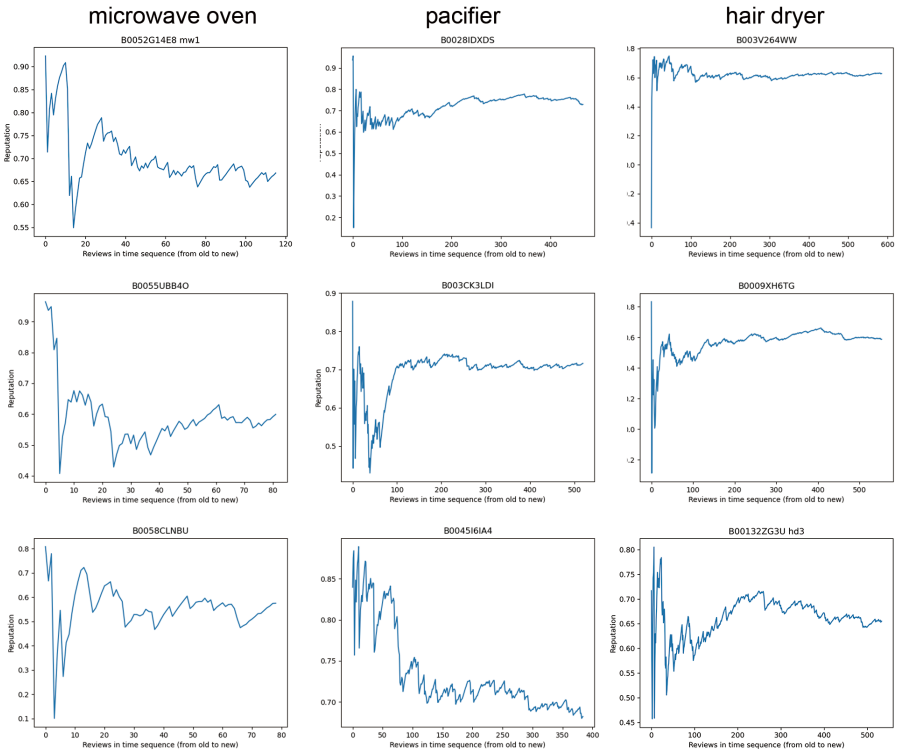


Figure 9: The reputation of 3 best-sold production(up to down) for microwave oven(left), pacifier(middle) and hair dryer(right). The x axis are from the oldest to the newest

From 9, after a short time dumping, the reputation converge a value quickly with some small fluctuations. Which meet our expectations.

## 4.3 Comprehensive scores

In order to analysis a potentially successful or failing product, we need to combine text-based measures and rating-based measures in a proper way. As we mentioned previously, we have formulated a total score to describe the overall evaluation to the product the consumers reviewed for each piece of review. Now, we combine text-based measures and rating-based measures as a measurement for each product. Moreover, such measurement may even vary in different time.

Since the volume of sales is highly related to the number of reviews, we can simply assume that we can use the number of reviews to represent the volume of sales. In addition, the volume of sales could be a convincing criterion for whether the product is successful or failing. In order to analysis the potentially successful or failing of the product, we need to focus on the changing of the volume of sales as time varies and figure out what cause the changing on sales of each product.

In order to figure out a measurement that well indicate a potentially successful or failing product based on the combinations of text-based measures and ratings-based measures, we need to analysis the changing of such measurement as time varies. More importantly, we need to compare such two time-series, the sales and the measurement we derived, and analysis their relationship.

Without loss of generality, we pick up some of the product and take one month as time periods. For each month, we compute the average total score which we have mentioned in previous evaluation system mainly based on the star rating and review contents. And, we also compute the total number of reviews representing the volume of sales of each month. Since the consumers will mainly focus on recent reviews while pay less attention to the previous review written in long time ago, we give more weights to the average total score of the current month and less weights to last month and compute their weighted average as the measurement which is shown as below:

$$0.3x_{i-1}y_{i-1} + \frac{0.7x_iy_i}{0.3y_{i-1} + 0.7y_i} \tag{4}$$

Then, we can plot such two lines and see the trends of such measurement and the sales on each month. More importantly, we can use Granger causality test to figure out whether such measurement is a good indication of the sales.

Table 1: Result of Granger test for product 1

| | | | | |
|---|---|---|---|---|
| Granger Causality | number of lags (no zero) 1 | | | |
| ssr based F test | F=4.6844 | p=0.0351 | df_denom=52 | df_num=1 |
| ssr based chi2 test | chi2=4.9547 | p=0.0260 | df=1 | |
| likelihood ratio test | chi2=4.7441 | p=0.0294 | df=1 | |
| parameter F test | F=4.6844 | p=0.0351 | df_denom=52 | df_num=1 |

Table 2: Result of Granger test for product 2

| | | | | |
|---|---|---|---|---|
| Granger Causality | number of lags (no zero) 1 | | | |
| ssr based F test | F=3.1784 | p=0.0805 | df_denom=52 | df_num=1 |
| ssr based chi2 test | chi2=3.3618 | p=0.0667 | df=1 | |
| likelihood ratio test | chi2=3.2631 | p=0.0709 | df=1 | |
| parameter F test | F=3.1784 | p=0.0805 | df_denom=52 | df_num=1 |

Granger causality test is one of the most efficient method to analysis the causal relationship between two time series. By applying Granger causality test, we can find out from the result that such measurement is indeed the indication of the sales, which means this measurement can well indicate whether the product is potentially successful or failing.

From the result of two kinds of product we pick up, we can see that one of the p-value is 0.03 and the other p-value is 0.06, which is very low. So, we can reject the hypothesis and accept that such measurement is useful in forecasting the volumes of sales. In other words, we can conclude from the results that such measurement is indeed the indication of the sales, which means this measurement can well indicate whether the product is potentially successful or failing

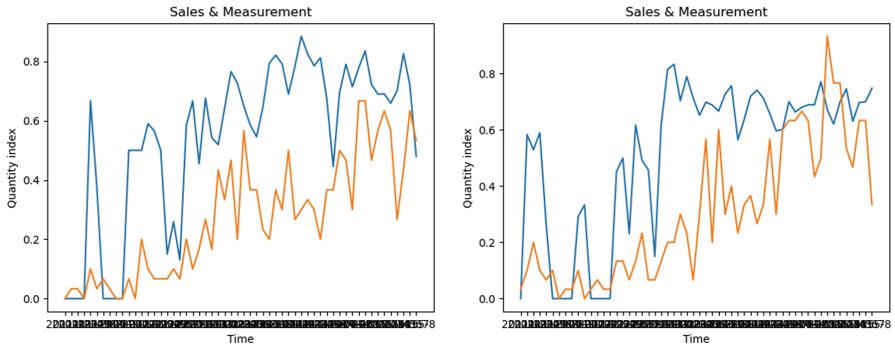


Figure 10: Sales & Measurement for product(left 1, right 2)

## 4.4 Recent star ratings effects

Suppose an Amazon user who just buys a pacifier and feels good when you receive it. He would like to vote a good stars, but he finds that the recent reviews are not good. His reactions may be dispersal. He might be a man of principle, might believe the product deserves more, might be influenced by bad votes. On the contrary, if he sees a series of good reviews, does he have a tendency to vote 5 star? How do we determine these features? Our model suggests a statistical method to solve this problem and comfort Sunshine Company that their potential costumers **are not likely** to be influenced by another users.

Because five stars reviews are most significant in both three cases. We just consider the feature of five stars. Further analysis can shows other stars is qualitative same as five stars. Suppose the distribution of 5 star rating obey Bernoulli distribution with probability p, that is:

$$S \sim Ber(p) \tag{5}$$

Center Limit Theorem suggests independent random variable:$S_1, ..., S_n$:

$$\sqrt{n}\frac{\bar{S}-p}{p(1-p)} \to \mathcal{N}(0,1) \tag{6}$$

So, when n is big enough:

$$P(|\sqrt{n}\frac{\bar{S}-p}{p(1-p)}| > x) = 2(1-\Phi(x)) \tag{7}$$

where $\Phi(x)$ is the accumulate distribution function of $\mathcal{N}(0,1)$

Define $q_{\alpha/2}$ with $\Phi(q_{\alpha/2}) = 1-\alpha$

So,

$$P(\bar{S} \in [p - q_{\alpha/2}\frac{p(1-p)}{\sqrt{n}}, p + q_{\alpha/2}\frac{p(1-p)}{\sqrt{n}}]) = 1-\alpha \tag{8}$$

Checking the normal distribution table we have $q_{0.05/2} = 1.96$ That is:

$$P(\bar{S} \in [p - 1.96\frac{p(1-p)}{\sqrt{n}}, p + 1.96\frac{p(1-p)}{\sqrt{n}}]) = 0.95 \tag{9}$$

, which is the confident interval. Donate by $XX_{95}$.

We count all three successive review whose sum <6(or average<2), and collection the next review. If the 5 star rating out of this interval, we have enough confidence to say that the customs are likely to be influence by a series bad reviews.

The numerical result is:

$$\begin{aligned} HD_{95} &= [0.549, 0.619] \\ MW_{95} &= [0.374, 0.452] \\ PA_{95} &= [.633, 0.711] \end{aligned} \tag{10}$$

$$\begin{aligned} P(HD_{bad}) &= 0.557 \\ P(MW_{bad}) &= 0.438 \\ P(PA_{bad}) &= 0.738 \end{aligned} \tag{11}$$

The result suggest that pacifier's costumers may more likely to rate more five star when they see a series bad review but no significant tendency can be found for hair dryer and microwave.

The abnormal behavior of pacifier can be understood. Because microwave oven and hair dryer are electric products. When we review a electric product, the functions and quality are always considered firstly. It seems that personal preference is not significant like pacifier. When one really like a pacifier, it is not strange that he wants to vote a better score than he expectation.

However, we should mention that, our result only suggests that statistical correlation is not significant. But for individuals, they may be influenced. For example, the "R3TQ3SO VQUMAZ0" review of hair dryer appears in a series of 2 star reviews. The guest say:"...I read the reviews about the hair dryer suddenly dying, but gambled that it wouldn't happen to me. Wrong. Turned it on this morning and nothing happened." Obviously, some reports of malfunction impact his opinion.

## 4.5 Descriptors of Reviews[5]


Figure 11: Word could of descriptors in good-rating (upper) and bad-rating (below) groups for microwave oven(left), pacifier(middle) and hair dryer(right)

We visualized the correlation between descriptors and ratings by depicting the word cloud of the descriptive adjectives from good rating (defined as a total score of above 0.5) and bad rating (defined as a total score of below -0.5). Figure 1 clearly demonstrate that positive descriptors such as "great", "perfect", "happy" are fairly common in the good-rating groups, while negative descriptors such as "old", "bad", "heavy" appear more often

in bad-rating groups. Moreover, each product is featured with its special functional descriptions, in the product microwave oven for instance, the "frozen" indicated customers may be concerned about the defrosting function of microwave ovens, and "Samsung" from the bad-rating group cast doubt on the quality of products from this brand, which inquires more in-depth investigations.

Notably, some positive descriptors also appear in the bad-rating group as shown in Figure 11. However, after a closer look at the review text, we attribute this abnormality to the semantic complexity, where people may use contrast (used to be good in the past but. . . .), sarcasm, complex-structured sentences to express their extreme unsatisfactory. Here is an example:

On 3/16/2015, a review (ID: R1Y4EFPGX0HDWH) wrote: "**Good** for 6 months, then DEAD! Was **great** until 6 months after received. It just stopped working. Trying to contact the manufacture. Will keep you posted on their customer service."

# 5 Conclusions

In this paper, we performed emotional sentiment analysis on text reviews to transform it to numerics, which serve as determinants together with helpfulness vote, star rating and date in evaluating a product and predicting the future market. Based on the model, we would be able to reflect product reputation, thus facilitating customers to compare different products to make purchase decision, indicating the success or fail of a product by conveying customers' opinion to manufactures, so that different marketing strategies could be launched when necessary; and proposing product design features based on keywords from the text review. However, the applicable scope of our model remain to be verified in practice.

# 6 Strengths and Weaknesses

**Strength**

1. Using BERT to perform word embedding and prepare inputs for RNN – accurate, reliable and time-saving.
2. Using helpfulness vote when sum weighted review score and star rating.
3. Using time related linear decay to weigh rating at different times – reflect recent opinion
4. Word cloud visualization of key words – intuitive and convincing.

**Weakness**

1. The effect of Vine is not carefully analyzed. We know that Amazon has Vine Voice, this indicate that a more close look at the information convey by vine customer may be helpful.
2. Non-vine customers who returned their purchases are not considered.
3. Competition between items may exist.

**Letter to the Marketing Director of Sunshine Company**

Dear Sir/Madam,
We are a team of students from 2020 MCM, wishing to offer some useful advice regarding your choice of product and prediction of market based on previous sell records. We are honored to present our multi-source evaluation model to your esteemed cooperation.

One of our basic ideas is to encompass the text review in our model, and the first issue to be addressed is unifying the form of all parameters – text reviews are consist of words, whose meaning are obscure to computers, while other information such as star ratings, helpfulness votes are composed of numbers, which can be some sort of 'understood' by computers. To convert text reviews as numbers that can reflect personal likes and dislikes (similar to star ratings), we used a deep-learning guided model. By training the deep learning model with large amounts of data that is already known and validated in advance, we can achieve the judgment of unknown data. However, as you can imagine, to train something from 'unknown' to 'predictive' requires huge amount of known data and is a highly time-consuming process, thus we employed the pre-trained BERT developed by Google, which is a very reliable transformer, to reduced our computational work. And the accuracy of the model after training reached 91.24%, which is high and reliable. Using this model, a score is generated according to the emotion embedded in each piece of text review (review_score). And to obtain a final score for each feedback, a natural notion is to sum weighted star rating and review_score. However, the helpfulness of each review is not invariable, thus changing the weight of review_score when helpfulness votes differ is a wise choice. Through this method, we get a rating for the each comment. To represent the reputation of product over time, a common method is using the meridian. However, the meridian fails to represent the recent ratings, which can cover more applicable information. To address to the problem, we introduced time related linear decay, which assigns weight to each ratings according to the interval between the date and the current time. Thus, a weighted meridian is better to represent product reputation. On the other hand, product success is further related to sales volume, with the same decay notion, we average weighted rating to reflect success. Moreover, in our analysis, no obvious relation between type of review and specific star ratings is identified, indicating that the customers are not easily influenced.

We want to specifically introduce our 'word cloud' visualization results, which can intuitively reflect customer's attitudes. And the word cloud visualization of descriptive words in high- (upper) and low- (below) rating groups were employed as an example here (Figure). The visualization result suggest that positive descriptors are more often presented in high-rating groups, while negative ones were often only found within the low-rating group. And the positive descriptors such as "good" found in the low-rating group are a result from semantic complexity. More importantly, the word cloud can provide information regarding product improvement. Take the frequently-seen word from microwave oven for example, "frozen" indicated that customers may be concerned about the defrosting function of microwave ovens, and "Samsung" from the bad-rating group casts doubt on the quality of products from this brand. With some in-depth inquires, the manufactural can easily know what the market demand is to adjust their sell strategy (eg: optimaze defrosting function, quality control of products from "Samsung").


Figure 12: Word could of descriptors in good-rating (upper) and bad-rating (below) groups for microwave oven(left), pacifier(middle) and hair dryer(right)

That's roughly all we have got in this study. Hopefully our job will helpful for future marketing of your esteemed company. All the best.

Sincerely yours
Team#2018901
MCM 2020

# Appendices

## Deep learning codes

```
import torch
import random
import numpy as np
import os
from transformers import BertTokenizer, BertModel
from torchtext import data
from torchtext import datasets
import torch.nn as nn
import torch.optim as optim
import time

SEED = 1234
random.seed(SEED)
np.random.seed(SEED)
torch.manual_seed(SEED)
torch.backends.cudnn.deterministic = True

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
init_token = tokenizer.cls_token
eos_token = tokenizer.sep_token
pad_token = tokenizer.pad_token
unk_token = tokenizer.unk_token
init_token_idx = tokenizer.cls_token_id
eos_token_idx = tokenizer.sep_token_id
pad_token_idx = tokenizer.pad_token_id
unk_token_idx = tokenizer.unk_token_id
max_input_length = tokenizer.max_model_input_sizes['bert-base-uncased']


def tokenize_and_cut(sentence):
    tokens = tokenizer.tokenize(sentence)
    tokens = tokens[:max_input_length - 2]
    return tokens


TEXT = data.Field(batch_first=True,
                  use_vocab=False,
                  tokenize=tokenize_and_cut,
                  preprocessing=tokenizer.convert_tokens_to_ids,
                  init_token=init_token_idx,
                  eos_token=eos_token_idx,
                  pad_token=pad_token_idx,
                  unk_token=unk_token_idx)
LABEL = data.LabelField(dtype=torch.float)
train_data, test_data = datasets.IMDB.splits(TEXT, LABEL)
train_data, valid_data = train_data.split(random_state=random.seed(SEED))
print(f"Number of training examples: {len(train_data)}")
print(f"Number of validation examples: {len(valid_data)}")
```

```
print(f"Number of testing examples: {len(test_data)}")
LABEL.build_vocab(train_data)
BATCH_SIZE = 32
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
if torch.cuda.is_available():
   print("CUDA")
else:
   print('CPU')
train_iterator, valid_iterator, test_iterator =
   data.BucketIterator.splits(
   (train_data, valid_data, test_data),
   batch_size=BATCH_SIZE,
   device=device)
bert = BertModel.from_pretrained("bert-base-uncased")


class BERTGRUSentiment(nn.Module):
   def __init__(self,
                bert,
                hidden_dim,
                output_dim,
                n_layers,
                bidirectional,
                dropout):
      super().__init__()
      self.bert = bert
      embedding_dim = bert.config.to_dict()['hidden_size']
      self.rnn = nn.GRU(embedding_dim,
                        hidden_dim,
                        num_layers=n_layers,
                        bidirectional=bidirectional,
                        batch_first=True,
                        dropout=0 if n_layers < 2 else dropout)
      self.out = nn.Linear(hidden_dim * 2 if bidirectional else
         hidden_dim, output_dim)
      self.dropout = nn.Dropout(dropout)

   def forward(self, text):
      # text = [batch size, sent len]
      with torch.no_grad():
         embedded = self.bert(text)[0]
      # embedded = [batch size, sent len, emb dim]
      _, hidden = self.rnn(embedded)
      # hidden = [n layers * n directions, batch size, emb dim]
      if self.rnn.bidirectional:
         hidden = self.dropout(torch.cat((hidden[-2, :, :], hidden[-1,
            :, :]), dim=1))
      else:
         hidden = self.dropout(hidden[-1, :, :])
      # hidden = [batch size, hid dim]
      output = self.out(hidden)
      # output = [batch size, out dim]
      return output
```

```
HIDDEN_DIM = 256
OUTPUT_DIM = 1
N_LAYERS = 2
BIDIRECTIONAL = True
DROPOUT = 0.25
model = BERTGRUSentiment(bert,
                         HIDDEN_DIM,
                         OUTPUT_DIM,
                         N_LAYERS,
                         BIDIRECTIONAL,
                         DROPOUT)


def count_parameters(model):
    return sum(p.numel() for p in model.parameters() if p.requires_grad)


for name, param in model.named_parameters():
    if name.startswith('bert'):
        param.requires_grad = False
print("{count_parameters(model):,} trainable parameters")
optimizer = optim.Adam(model.parameters())
criterion = nn.BCEWithLogitsLoss()
model = model.to(device)
criterion = criterion.to(device)


def binary_accuracy(preds, y):
    """
    Returns accuracy per batch, i.e. if you get 8/10 right, this returns
        0.8, NOT 8
    """

    # round predictions to the closest integer
    rounded_preds = torch.round(torch.sigmoid(preds))
    correct = (rounded_preds == y).float() # convert into float for
        division
    acc = correct.sum() / len(correct)
    return acc


def train(model, iterator, optimizer, criterion):
    epoch_loss = 0
    epoch_acc = 0

    model.train()
    #counter = 0
    for batch in iterator:
        optimizer.zero_grad()

        predictions = model(batch.text).squeeze(1)
```

```
        loss = criterion(predictions, batch.label)

        acc = binary_accuracy(predictions, batch.label)

        loss.backward()

        optimizer.step()

        epoch_loss += loss.item()
        epoch_acc += acc.item()
        #counter += 1
        #print((counter*BATCH_SIZE)/17500)
    return epoch_loss / len(iterator), epoch_acc / len(iterator)


def evaluate(model, iterator, criterion):
    epoch_loss = 0
    epoch_acc = 0

    model.eval()

    with torch.no_grad():
        for batch in iterator:
            predictions = model(batch.text).squeeze(1)

            loss = criterion(predictions, batch.label)

            acc = binary_accuracy(predictions, batch.label)

            epoch_loss += loss.item()
            epoch_acc += acc.item()

    return epoch_loss / len(iterator), epoch_acc / len(iterator)


def epoch_time(start_time, end_time):
    elapsed_time = end_time - start_time
    elapsed_mins = int(elapsed_time / 60)
    elapsed_secs = int(elapsed_time - (elapsed_mins * 60))
    return elapsed_mins, elapsed_secs


N_EPOCHS = 3

best_valid_loss = float('inf')
print("Start Train")
for epoch in range(N_EPOCHS):

    start_time = time.time()

    train_loss, train_acc = train(model, train_iterator, optimizer,
        criterion)
```

```
    valid_loss, valid_acc = evaluate(model, valid_iterator, criterion)

    end_time = time.time()

    epoch_mins, epoch_secs = epoch_time(start_time, end_time)

    if valid_loss < best_valid_loss:
        best_valid_loss = valid_loss
        torch.save(model.state_dict(), 'nlp-model.pt')

    print(f'Epoch: {epoch + 1:02} | Epoch Time: {epoch_mins}m
        {epoch_secs}s')
    print(f'\tTrain Loss: {train_loss:.3f} | Train Acc: {train_acc *
        100:.2f}%')
    print(f'\t Val. Loss: {valid_loss:.3f} | Val. Acc: {valid_acc *
        100:.2f}%')

test_loss, test_acc = evaluate(model, test_iterator, criterion)
print(f'Test Loss: {test_loss:.3f} | Test Acc: {test_acc*100:.2f}%')
```

# References


[1] bentrevett. pytorch-sentiment-analysis. https://github.com/bentrevett/pytorch-sentiment-analysis, 2019.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[3] Umar Farooq, Antoine Nongaillard, Yacine Ouzrout, and Muhammad Abdul Qadir. A multi source product reputation model. *Computers in Industry*, 83:55 – 67, 2016.

[4] Florent Garcin, Boi Faltings, and Radu Jurca. Aggregating reputation feedback. *Proceedings of the First International Conference on Reputation: Theory and Technology*, 1(1):62–74, 2009.

[5] louiefb. amazon-reviews-nlp. https://github.com/louiefb/amazon-reviews-nlp, 2019.