Problem Chosen
C

2020 MCM/ICM Summary Sheet

Team Control Number
2022378

Interpreting Customer Feedback in Online Market

Summary

We have provided several effective models to extract useful information from Amazon reviews of hair dryers, microwave ovens and baby pacifiers.

We started with several assumptions. We assume that the sales of a product is proportional to the number of reviews. We assume that the number of helpfulness votes is roughly proportional to time. We assume that customers tend to vote more helpful votes if a review already have many helpful votes, so the marginal importance of each votes should decrease. We assume verified buyers and vine customers reviews are more important. We assume if a review receives both many helpful votes and unhelpful votes, the review is considered as controversial and therefore less reliable. We assume that the data provided is the latest.

With these assumptions we first developed a general model of reviews for later use, which we refer to as the standard model of review. The general model of reviews takes into consideration the number of helpful votes, the number of unhelpful votes and the identity of the reviewer and combined the information together according to the assumptions. Then we built other models based on the standard model of review.

We developed the importance of review model to identity a stable and effective measure (Equation for the informativeness of reviews by combining the standard model with the length of the review. The model assigns a score for each review. If a review scores higher, the review contains more information. We then developed the brand reputation model to determine the reputation of each brand for a specific month, the result of which then can be used to determine the reputation trend for each brand. A third model called successful feature model is also developed to extract both good and bad features from the products of rising and falling brands in terms of their reputations.

We then analyzed the provided Amazon review data using our models. We showed that our models are stable in terms of the selection of parameters. We also showed that our models represents realistic measures for each of the properties of the reviews by manually analyzing some results from our model. We also identified whether some star ratings can induce more reviews and it turned out that low star ratings can induce high rating reviews in some degree. We found that some descriptive words are of high correlation with the star rating. In addition, we identified some target customers and the fluctuation in the number of sales in different months in a year, from which, Sunshine Company could develop sales strategy.

Although our models are both strong and effective, we hope to improve it in the future with some machine learning techniques and more data, so that we can provide even stronger models.

Letter to the Sunshine Company

Dear Marketing Director of Sunshine Company,

It is a great honor to have such a great opportunity to work with Sumshine Company. We understand that Sunshine Company value the customers and always tries the best the satisfy all customers. Therefore, we believe that previous customer reviews are crucial for both the developments and the future improvements of a product. We hope that our proposed measures and the analysis of the Amazon reviews can help your company to produce the best products for your customers.

While we are far from providing a comprehensive analysis for the market trends, our central results can provide some insights for measures of customer reviews, potential important design features and sales strategies for each product. We designed data measures for filtering most informative data, visualizing product's reputation, and predicting the potential success or failure of a product.

By extracting the key words in the text-based reviews, our models identify multiple important design features that the company might want to consider during the design process. The potential important design features for hair dryers are "core" which indicates the customers prefer longer cores or cores that do not tangle, "retractable", and "powerful". For microwave ovens, the important design features are "space" which indicates the customers prefer microwave ovens with larger space inside, "easy", and "stainless". For baby pacifiers, the important design features are "cute", "giraffe" and "monkey" which indicates the customers prefer pacifiers with giraffe and monkey shape.

Our analysis has also implied several potential sales strategies for market participation. Our models indicate that the characteristics of target customers of hair dryers should be female, young-age (<18 years old) and middle-aged (18–60 years old). The suggestion for timing of market participation is December, January and February. For microwave ovens, male customers have a relatively higher standard for microwave ovens, and the Company should focus more on improving the microwave functions that would be appreciated by male customers. The target customers for microwave ovens are primarily young people and middle-aged people. The suggestion for timing of market participation is summer and winter. For baby pacifiers, the target users are young people. The number of sales shoot up in January, July, August and December. Therefore, the company should prepare more stock during these months.

We truly hope this report will provide an insightful analysis for market participation choices and product design feature choices. We wish you the best in your future endeavors.

Sincerely yours

MCM 2020 Team 2022378

Contents

1 Introduction

1.1 Background

Amazon, as one of the biggest online marketplaces in the U.S., provides a convenient platform for buyers to communicate with other potential customers through the rating and review system. In the system, buyers of a product can express whether they are satisfied with the product by rating it with stars 1 to 5 (1 stands for the lowest satisfaction and 5 stands for the highest satisfaction). Meanwhile, buyers can also post reviews, which are text-based messages, to indicate further details. Other potential customers can vote a review as helpful or unhelpful based on whether they agree with the review or not. All the data here would be valuable and insightful for companies to understand the market trend, design the products, and formulate the sales strategy.

Recently, Sunshine Company has interest in producing and introducing a microwave oven, a baby pacifier, and a hair dryer in the online marketplace. As the consultants of Sunshine Company, our job is to generate an online sales strategy and identify the potentially successful design features that company might want to include in their own designs based on the ratings and reviews of competing products as well as our model analysis.

1.2 Problem Restatement

Sunshine Company has provided the rating and review data sets of other competing products. We will analyze the data sets using the mathematical models we built based on real life circumstances. Our goal is to identify the successful product features and formulate the sale strategy by addressing meaningful quantitative and/or qualitative patterns, correlations, measures, and parameters within and between star ratings, reviews, and helpfulness ratings. In particular, we will provide our approaches for solving different problems as following:

- We will provide a mathematical model called Importance of Review Model (IR Model) for the company to filter out the most informative reviews to track once the three products are launched online.
- We will provide a mathematical model called Brand Reputation Model (BR Model) to measure and identify the increasing or decreasing pattern for each product's reputation along with time.
- We will provide a mathematical model called Successful Feature Model (SF Model) to measure and identify the potentially important product features based on text reviews and ratings.
- We will demonstrate the correlation between the existing star ratings and the buyers' motivations of writing new reviews.
- We will specify the correlation between specific quality descriptors in the reviews and the rating levels.

1.3 Assumptions

We make the following basic assumptions in order to simplify the problem. Each of our assumptions is justified and is consistent with the basic fact.

- **The sales of a product is proportional to the number of reviews it received.** It is a common sense that a better selling product will receive more reviews in the online market.
- **The number of total helpfulness votes is roughly proportional to the function of time.** Earlier reviews have a longer time to gather helpfulness votes. This assumption will be used in the Standard Model.
- **Customers will tend to vote a review as helpful if the review has already received many helpful votes, and the situation for unhelpful vote should be similar.** According the group psychology, the Bandwagon effect suggest that the increasing rate of trends will be higher if the trends have already been adopted by others [1]. This assumption will be used in the Standard Model.
- **The reviews written by vine and verified customers should be more reliable with less emotional effects.** Therefore, if a review is written by a verified or vine customer, the review should have a higher weight. This assumption affects our decision of choosing parameters.
- **If a review receives many votes for both helpfulness and unhelpfulness, this review is not representative and should be consider less reliable.** If a review is controversial, the information in the review should be less considered. This assumption will be used in the Standard Model.
- **The data provided by the company is the latest data from Amazon by the time our team was hired.** The helpful votes are accumulated till August 31st, 2015.

2 Models Construction

2.1 Notations

Symbol Name	**Description**	**Notes**
$\mathcal{R}$	The set of all reviews.	Subscripts can indicate subsets.
r	A specific review.	$r \in R$.
$\mathbb{M}$	Standard model of review.	A model used extensively.
N_{tot}	Total number of votes for a review.	A function of r.
N_+	Number of helpful votes for a review.	A function of r.
N_-	Number of unhelpful votes for a review.	A function of r.
T	Number of months since the review was posted.	A function of r.
L	Number of characters in a review.	A function of r.
S	Number of stars for a review rating	A function of r.
W	The weight of a review	A function of r. To be defined later.
V	Whether the review is a from a verified buyer.	A function of r. $V(r) = 1$ if the reviewer is verified and $V(r) = 0$ otherwise.
E	Whether the review is from a vine customer.	A function of r. $E(r) = 1$ if the reviewer is vine and $E(r) = 0$ otherwise.
$\alpha, \beta, \gamma, \delta, \cdots$	Adjustable model parameters.	Different models have different parameters. Subscription indicates the specific model.

Table 1: Symbols used in models.

2.2 Standard Model of Reviews

This model will be used as a sub-model for the first three models. The model reads

$$\log_\alpha\left(\frac{N_+}{T} + \alpha\right)\beta^{\frac{N_-}{T}}\gamma^V\delta^E. \tag{1}$$

The model can be divided into several parts.

The first part takes the helpful votes of a review into account. We model this part with a function containing one adjustable parameter α, which for obvious reasons is chosen to be $\alpha > 1$. We divide N_+ with T because our assumption states that the number of votes for a specific review should be roughly proportional to the time it has existed. We add α to the equation so that the value $\log_\alpha\left(\frac{N_+}{T} + \alpha\right)$ is evaluated to 1 when there has no helpful vote. This function is a strictly increasing, concave down function with respect to N_+/T. From our assumption, customers tend to vote more helpful for a review if there are already a lot of helpful votes, so the marginal importance of each vote should decrease and therefore the function should be concave down. The first part of the function also passes through the point $(0, 1)$, since if there is no vote, the score should not be affected.

The second part considers the unhelpful votes. This part also has a adjustable parameter and reads $\beta^{\frac{N_-}{T}}$. As the previous part, N_- is also divided by T. Clearly, $\beta < 1$, since unhelpful votes should have negative impact on the review.

We have assumed that a review receiving large number of both helpful votes and unhelpful votes is less reliable compared to a review receiving the same proportion of helpful votes and unhelpful votes. One of the corollary of this assumption is that if $N_+ = N_- = \frac{N_{tot}}{2}$, then $\log_\alpha\left(\frac{N_{tot}}{2T} + \alpha\right)\beta^{\frac{N_{tot}}{2T}}$ should be strictly decreasing with respect to $N_{tot}/2T$. We can test the model for this assumption by choosing several values of α and β. Without losing any generality, let's choose $\alpha = e$. Bellow are the plots for $\ln(x + e)\beta^x$ with different values of β.

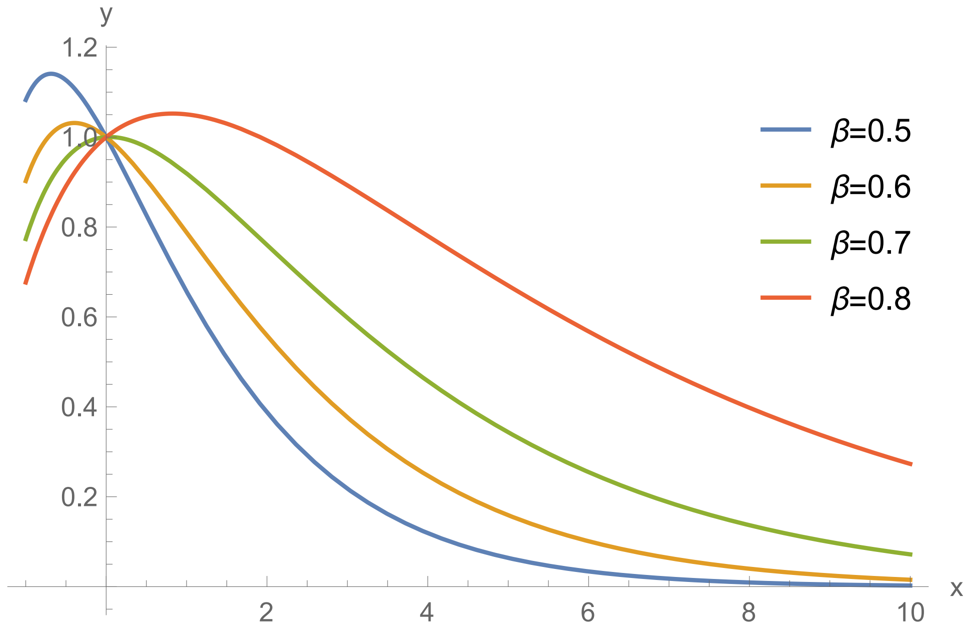

Figure 1: $y = \ln(x + e)\beta^x$ for different values of β.

It can be noted that for $\beta = 0.5$ and $\beta = 0.6$, the function is strictly decreasing for all $x > 0$, but for $\beta = 0.8$, it's not. Since we want a decreasing function for all $x > 0$, but we don't want the function to decrease too fast. It is nice to ask for the derivative of the function to be zero when $x = 0$ so that the function will remain decreasing for all $x > 0$. Taking the derivative at $x = 0$ and setting it to 0, we have

$$\frac{d}{dx}\left[\log_\alpha(x + \alpha)\beta^x\right]_{x=0} = \frac{1}{\alpha}\ln(-\alpha) + \ln\beta = 0. \tag{2}$$

Solve the equation we have

$$\beta = e^{-\frac{1}{\alpha}\ln\alpha}. \tag{3}$$

With this constraint, we have successfully reduced the number of free parameters by one. Now, let's check the function $\log_\alpha(x+\alpha)\,\beta^x$ under the constraint above for different values of α.

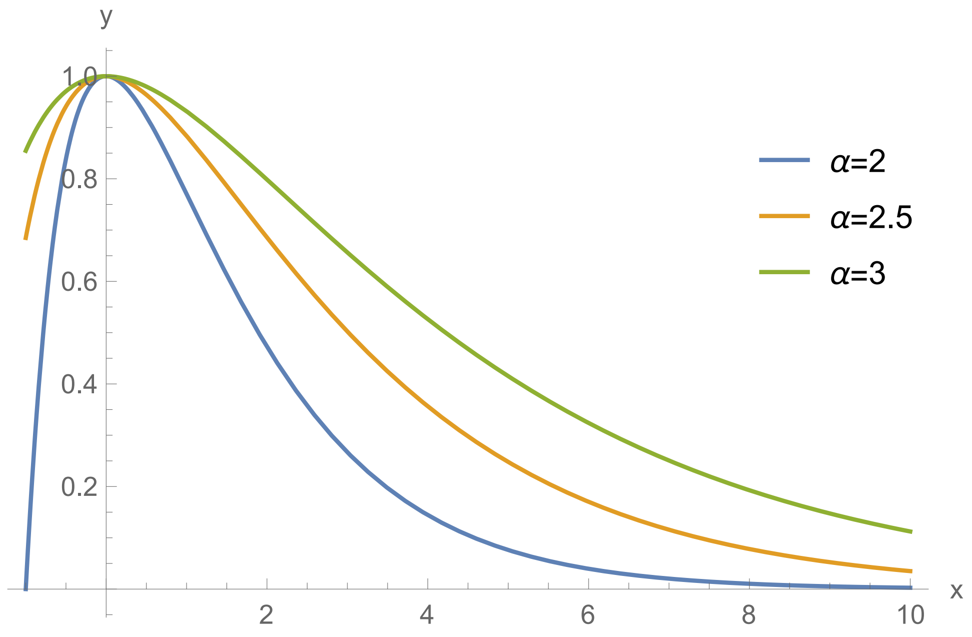

Figure 2: $y = \log_\alpha(x+\alpha)\left(e^{-\frac{1}{\alpha}\ln\alpha}\right)^x$ for different values of α.

In deed, as expected, the function decreases for all $x > 0$ and we can adjust the parameter to decide how fast the function decreases.

The next two parts of the model, γ^V and δ^E, are quite self-explanatory. Both V and E takes only 0 and 1, so raising them to the exponents of some parameters would result in the model being multiplied by the parameters or 1. We require both $\gamma, \delta > 1$ since we assume that a verified user's review is more important than a non-verified user's review. Similarly, a vine user's review is also more important than a non-vine user's review.

Putting everything together, we have the standard model of review, which we will refer to as the standard model,

$$\boxed{\mathbb{M} \equiv \log_\alpha\left(\frac{N_+}{T}+\alpha\right)\beta^{\frac{N_-}{T}}\gamma^V\delta^E}, \tag{4}$$

under the constraints

$$\boxed{\begin{aligned} &\alpha > 1, && (5)\\ &\beta = e^{-\frac{1}{\alpha}\ln\alpha}, && (6)\\ &\gamma > 1, && (7)\\ &\delta > 1. && (8)\end{aligned}}$$

To be noted that N_+, N_-, T, V and E are all functions of reviews. They would take a specific review as an argument and returns the corresponding information about the review (number of helpful votes, number of unhelpful votes, etc.) So the standard model is also a function of review, i.e.

$$\boxed{\mathbb{M}(r) = \log_{\alpha}\left(\frac{N_{+}(r)}{T(r)} + \alpha\right)\beta^{\frac{N_{-}(r)}{T(r)}}\gamma^{V(r)}\delta^{E(r)}}. \tag{9}$$

2.3 Importance of Review Model (IR Model)

This model aims to identify the most important reviews. The model assigns a score for each review based on its length, number of helpful votes, number of unhelpful votes, and some other information. The score determines the informativeness of a review. A higher score review is more informative and a lower score review is less informative.

This model used the standard model $\mathbb{M}$ and multiplies it with the natural log of L, the length of the review. It is self-evident that the longer a review is, the more information the review should contain, but if a review is extremely long, the marginal information of the review can decrease as its length increases, so we model this part with a logarithm function. Additionally, other properties of a review can also suggest the review's informativeness. Since other properties are already encoded in the standard model, we simply weight each review using the standard model.

In mathematical notation, let $\mathcal{I}$ be the set of scores measuring the importance of each review, we have the model

$$M_{IR} : \mathcal{R} \to \mathcal{I}, \tag{10}$$

such that

$$M_{IR}(r) \equiv \ln L(r)\mathbb{M}_A(r) = \ln L(r)\log_{\alpha_A}\left(\frac{N_{+}(r)}{T(r)} + \alpha_A\right)\beta_A^{\frac{N_{-}(r)}{T(r)}}\gamma_A^{V(r)}\delta_A^{E(r)}, \tag{11}$$

where the subscript A means that the set of parameters can only be applied to this model.

Interpretation

The score for each review measures the importance of the review. Therefore, the reviews scoring the highest values are the most informative models and the company should therefore focusing one these reviews.

2.4 Brand Reputation Model (BR Model)

This model aims to determine the reputation of a model. The model assigns a score for each brand based on its reviews. The model weights each review based on the author of the review (verified or not, vine or not), and the helpfulness rating. Theoretically, a review written by a verified or vine customer with a high helpfulness rating will have a higher weight.

Definition 1. ***Criteria of Brand Reputation.*** The reputation of a brand is defined by the sum of its weighted.

To assign a weight for each reviews, we note that a review with rating 4 or 5 should have a positive effect on a brand's reputation, yet a review with rating 1 or 2 should have a negative effect on a brand's reputation. A review with rating 3 has an ambiguous altitude toward the brand and therefore should not contribute to the brand's reputation. Therefore, the first part of the model is $S - 3$.

The number of positive votes, the number of negative votes and other properties of a review is already encoded in the standard model, so we calculate the weight using the formula

$$W(r) \equiv (S(r) - 3)\mathbb{M}_B(r) = (S(r) - 3) \log_{\alpha_B} \left(\frac{N_+(r)}{T(r)} + \alpha_B \right) \beta_B^{\frac{N_-(r)}{T(r)}} \gamma_B^{V(r)} \delta_B^{E(r)}, \tag{12}$$

where, again, the subscript B means the parameters only applies to this model.

After determining the weight of each review, we need to add up the weights for the same brand. Let $\mathcal{B}$ be the set of all brands. Then, define $\mathcal{R}_{b,i}, \forall b \in \mathcal{B}$ be the set of reviews for each brand for the ith month. Let $\mathcal{P}_i$ be the set of scores measuring the reputation of each brand for the ith month. We have the model

$$M_{BR} : \mathcal{B} \longrightarrow \mathcal{P}_i, \tag{13}$$

such that

$$\begin{aligned} M_{BR}(b, i) &\equiv \sum_{r \in \mathcal{R}_{b,i}} W(r) \\ &= \sum_{r \in \mathcal{R}_{b,i}} (S(r) - 3)\mathbb{M}_B(r) \\ &= \sum_{r \in \mathcal{R}_{b,i}} (S(r) - 3) \log_{\alpha_B} \left(\frac{N_+(r)}{T(r)} + \alpha_B \right) \beta_B^{\frac{N_-(r)}{T(r)}} \gamma_B^{V(r)} \delta_B^{E(r)}. \end{aligned} \tag{14}$$

Interpretation

The model assigns a score for each brand in each month. The brands scoring the highest values for a month is considered to have the best reputation for this month and vice versa. To be noted that the score is positive for brands with good reputations and negative for brands with bad reputations. Then, for each brand, the reputation score can be plotted with respect to date to determine whether a brand's reputation is increasing or decreasing.

We want to extend this model to compute the reputation of each word that appeared in the review text.

Definition 2. ***Criteria of Word Reputation.*** The reputation of a word is defined as the sum of weighted reviews in which the word appears.

We use the same function above to calculate the weight of a given review. For each word in a given review, we add the weight of the given review to each word. Therefore, the reputation of an arbitrary word is the sum of all review weights that the word appears in. Let $\mathcal{W}$ be the set of all different words in all the reviews. Then, define $\mathcal{T}_w$ be the set of reviews that contains the word w. Let Q_w be the score measuring the reputation of the word w. We define the word reputation model

$$M_{WR} : \mathcal{W} \longrightarrow Q_w, \tag{15}$$

such that

$$\begin{aligned} M_{WR}(w) &= \sum_{r \in \mathcal{T}_w} W(r) \\ &= \sum_{r \in \mathcal{T}_w} (S(r) - 3) \log_{\alpha_B} \left(\frac{N_+(r)}{T(r)} + \alpha_B \right) \beta_B^{\frac{N_-(r)}{T(r)}} \gamma_B^{V(r)} \delta_B^{E(r)}. \end{aligned} \tag{16}$$

Interpretation

The model assigns a score for each word that appears in the dataset. If the review has a high reputation and contains the word w, then w indicates a feature in some high reputation brands. On the other hand, if the review has a low reputation and contains the word w, then w indicates a feature in some low reputation brands.

2.5 Successful Feature Model (SF Model)

This models aims to extract the successful features from brands with the best reputations. In order to simplify the problem, some constrains are added to the definition of a successful brand based on the reputation of that product.

Definition 3. ***Criteria of Success.*** A successful brand is either

1) a brand with high average reputation over a certain time period, or

2) a brand with an increasing reputation pattern.

The BR model already computed the reputation scores for each brand for each month, so we need a model to determine the brands with best reputation. Let n be the total number of months, we can score each brand with two formulae.

$$P_1(b) = \frac{1}{n} \sum_{i=1}^{n} M_{BR}(b, i). \tag{17}$$

This formula is just an average of reputation.

$$P_2(b) = \frac{1}{n} \sum_{i=1}^{n} M_{BR}(b, i) + \frac{\omega}{n} \left[M_{BR}(b, n) - M_{BR}(b, 1) \right], \tag{18}$$

where ω is an additional adjustable parameter. The second formula takes into account whether the reputation is increasing or decreasing. The first part of this formula averages over all months to calculate the average reputation of a brand. The second part calculates the average increase in each brand's reputation. ω weights the average increase so that both part can be equally important.

Let $\mathcal{B}_{t,j}$ be the sets consisting only 5 brands that has the best reputation according to the formula P_j. Let $\mathcal{R}_{\mathcal{B}_{t,j}}$ be the set of reviews only for the brands in $\mathcal{B}_{t,j}$. Let $\mathcal{W}_j$ be the set of all words in the reviews $r \in \mathcal{R}_{\mathcal{B}_{t,j}}$. Let $\mathcal{R}_{w,j}$ be the set of reviews containing a specific word $w \in \mathcal{W}_j$ Then, we can weight the appearance of each word according to the corresponding review using the same weighting formula as the BR Model (Equation 12). Then, let $\mathcal{S}_j$ be the set of scores measuring the importance of each word (successfulness of each feature). We have the model

$$M_{SF} : \mathcal{W}_j \to \mathcal{S}_j, \tag{19}$$

such that

$$M_{SF}(w) \equiv \sum_{r \in \mathcal{R}_{w,j}} W(r), \tag{20}$$

where $W(r)$ is Equation 12.

In addition, we can also find the features of the brands with worst overall reputations. We only need to adjust the model such that the set $\mathcal{B}_{t,j}$ only includes the worst 5 brands according to their reputations.

Interpretation

Each word technically doesn't corresponding to each feature. There are many unimportant words such as "this", "am", "is", etc., bu it is easy to manually remove them. If we consider the remaining words to represent features of each brand, then the top scoring words represents the most successful features.

3 Model Extension and Analysis

3.1 Problem a: Most Informative Data Measures

To construct the most informative data measures, we first separated all the reviews into three groups according to the products. Then, we extracted the length, the helpfulness rating, the review's author status (vine or not, verified or not), and the review date from each review. We used the IR Model by applying the data stated above. The outcome of the IR Model is the score (weight) for each individual review. Here we chose α as e, γ as 1.1, and δ as 2 in order to balance the weight between different impacting factors. The results are following:

Product	**Score**	**Rating**	N_+	N_{tot}	**Vine**	**Verified**	**Review Body**	**Review Date**
Hair dryer	23.08	4	2	2	Y	N	Review 1 bellow	8/15/2015
Hair dryer	22.10	4	224	230	N	Y		5/10/2014
Hair dryer	21.56	3	46	56	Y	N		11/24/2014
Microwave	36.69	5	116	122	Y	N	Review 2 bellow	1/1/2015
Microwave	35.34	3	182	192	Y	N		1/8/2015
Microwave	21.33	5	104	117	Y	N		1/12/2015
Pacifier	23.58	1	29	29	N	N	Review 3 bellow	8/30/2015
Pacifier	19.08	5	5	5	Y	N		5/27/2015
Pacifier	190.1	4	283	284	N	N		7/23/2013

Table 2: Overlapping rate corresponding to each product

From the table, we found that most of the top score reviews are written by vine customers. By assumption, the reviews written by vine customers should be more pertinent. Most of the top score reviews also have high helpfulness rating, which means they can provide more information and they are approved by other customers. The review dates are all relatively closed to the present time (within two years), so that the reviews are representing the most updated feedback.

Due to the space limitation, we only present the text-based review for each product that receives the highest score:

- The highest score review for hair dryer: This is a very good blow dryer. It has three heat and two speed settings, and a cool shot button to help set your style. It's 1875 watts- so it's nice and strong though not as powerful my professional hair dryer that does dry my hair faster. But my hair has been getting split ends and damaged, so I love that this dryer is designed to protect against heat damage. Only time will tell if that feature works, but I sure need it. This also has the usual attachments that I never use but they look the same as what typically comes with a blow dryer...(220 words left)
- The highest score review for microwave: Before this microwave found its place on my counter this week, I hadn't known how much I would love it! It heats food quickly, is easy to use (once you review the manual), and can serve the same functions a microwave and toaster oven do. What makes this microwave different... It has many possibilities. Pizza is crispy on the bottom (when reheated) even on the microwave setting! It can be simple, or you can dive into all the stuff it can do! The size is comparable to our old 1.1 cu ft microwave, yet it can do more...(807 words left)
- The highest score review for pacifier: These bags make it obvious why the lansinoh bags are so popular. cons: the medela bags are nearly impossible to get zip closed, and i can never seem to do it without getting milk on the top and air inside the bag. they are also horrible for measurement. what looks like 5oz on the bag is actually around 4oz if you measure with a medela bottle... (231 words left)

By manually evaluating the top score reviews, our team found that all the top score reviews are pertinent, descriptive, logical, and informative. For example, from the top review of hair dryer, we can see that the author stated both cons and pros to make an argument, in which the company can quickly distinguish the bad features and good features from the review. Some of the top reviews contain several follow-up updates, in which the company can easily improve the after-sales service. Therefore, the top score reviews are coherent with the model prediction, which also prove the accuracy of out IR Model.

3.1.1 Stability Analysis

The variation of parameters in IR Model will consequently lead to the variation of importance ranking since the impacting factors are now weighted differently. However, if the IR Model is stable, a minor change in parameters will not affect the results much. Since β is depend on α, varying α is equivalent to vary β. Also, most of the customers are verified, thus changing γ will not affect on the relative score of informativeness. Similarly, only a few customers are vine, therefore, changing δ will not affect on the relative score of informativeness as well. Based on this logic, we choose to vary only α for stability test. Here, we need to define an evaluation criteria for the stability of our IR Model.

Definition 4. ***Overlapping Rate.*** Let $\mathcal{R}_{\alpha_A}$ be the set of first n reviews based on each review's score of informativeness. Let

$$u = \left| \bigcap_{k=1}^{m} \mathcal{R}_{\alpha_{A,k}} \right| \tag{21}$$

be the number of element in the union of $\mathcal{R}_{\alpha_A}$ for m different α_A's. Then, we define the overlapping rate

$$\sigma = \frac{u}{n}. \tag{22}$$

For testing, we choose $n = 100$, α as 2.5, e, and 3 respectively. The overlapping rate for each product is shown below:

Product	**Overlapping Rate**
Hair dryer	87.0%
Microwave	90.0%
Pacifier	89.0%

Table 3: Overlapping rate corresponding to each product

From the table, we see that the overlapping rates for all three products are all considerably high ($\geq 80\%$). Therefore, our IR Model passes the stability test.

At the mean time, since the IR Model is an extension from the Standard Model, it also means that the Standard Model passes the stability test, as well as the BR Model and SF Model, which are also extended from the Standard Model.

3.2 Problem b: Product's Reputation Time-based Measures

Figure 3 presents the time based reputation trend for some product. Here we chose α as e, γ as 1.1, and δ as 5 in order to balance the weight between different impacting factors. From top to bottom, the figure presents 2 ways of measuring the overall reputation, unweighted (P_1, Equation 17) or weighted (P_2, Equation 12) for each product. From left to right, the figure presents the first, second and last brand in terms of its reputation. Since the second way of measuring the overall reputation takes into account whether the trend is increasing or decreasing, we focus on the second, fourth and sixth row of the figure. It is clear that the highest ranking brand and the second highest ranking brand has an increasing reputation, whereas the lowest ranking brand has a decreasing reputation.

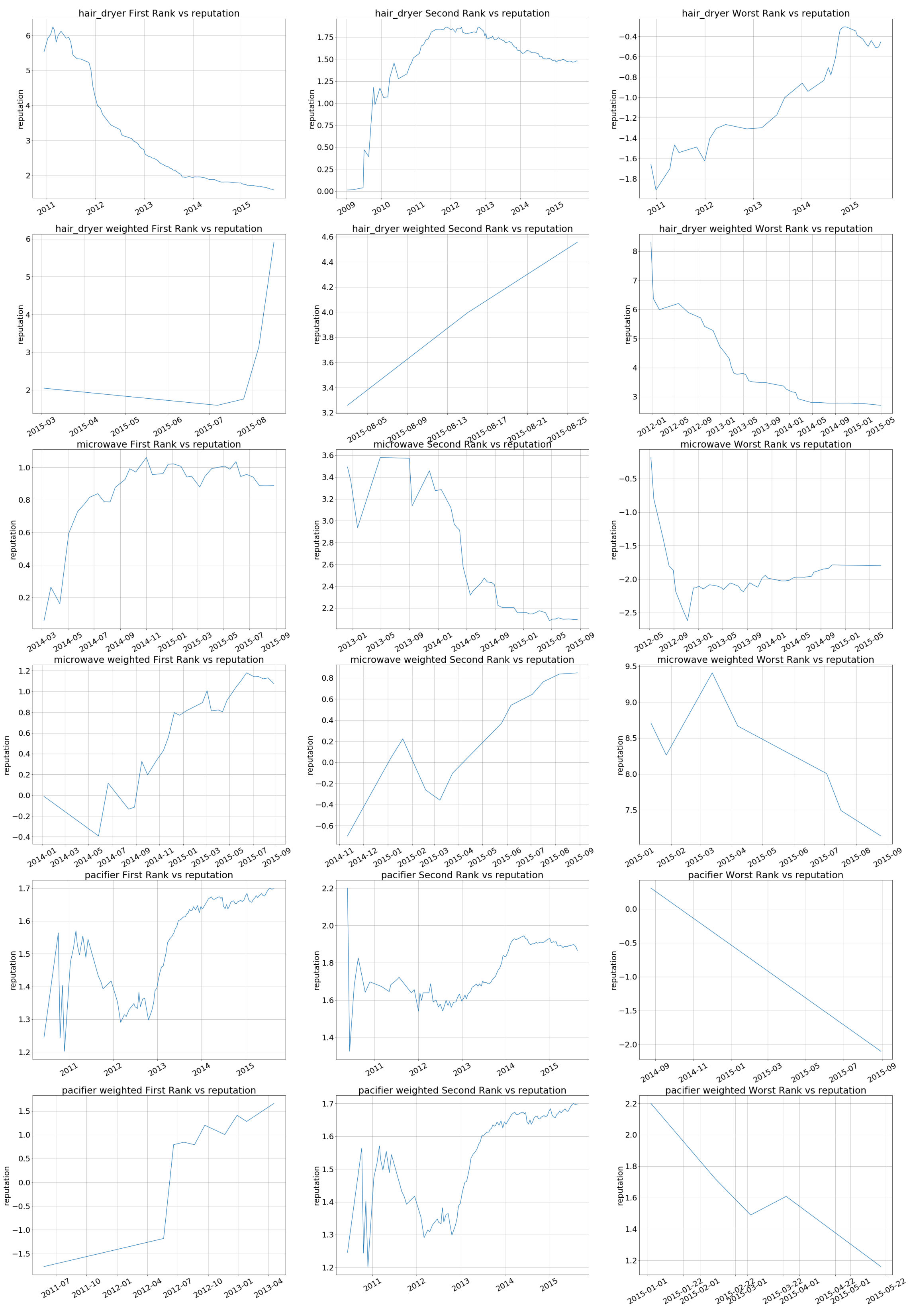

Figure 3: Brand reputation vs time graphs.

3.3 Problem c: Potential Success and Failure Data Measures

We want to use the word reputation model to find the words that have the highest weights and the words that have the lowest weights. For a given word, higher weight implies that the customers prefer the feature that the word describes and lower weight implies that the customers don't like the feature that the word describes. For simplicity, only words that describe characteristics of a given product are considered. When considering about the design of a given product, we can give importance to features that are described by high-weighted words and eliminate features that are described by low-weighted words. We use **WR Model**(the extension of **BR Model** in section 2.4) to evaluate the weight of each word. Here we chose α as e, γ as 1.1, and δ as 2 in order to balance the weight between different impacting factors. To enhance credibility, we only use the reviews from products with the top-10 highest reputations computed from the previous section for good words, and use the reviews from products with the 10 lowest reputations computed from the previous section for bad words. Table 4 indicates the 3 highest-weight words and some good features that are useful when designing each product, and Table 5 indicates the 3 lowest-weight words and some bad features that should be eliminated when designing each product.

Product	**Key Word**	**Rank**	**Weight**	**Interpretation**
Hair dryer	cord	1	60406.64	long cord/cord does not tangle
Hair dryer	retractable	2	19056.54	customers prefer retractable hair dryers
Hair dryer	powerful	3	12582.18	powerful hair dryer dries hair quickly
Microwave	space	1	3770.78	microwave has large space inside
Microwave	easy	2	3137.81	microwave is easy to use
Microwave	stainless	3	1540.86	customers prefer stainless microwaves
Pacifier	cute	1	31595.73	the shape of the pacifier is cute
Pacifier	giraffe	2	26673.61	babies like giraffe shape
Pacifier	monkey	3	23414.52	babies like monkey shape

Table 4: 3 highest-weight words and implied features

Product	**Key Word**	**Rank**	**Weight**	**Interpretation**
Hair dryer	heat	1	-8883.63	not enough heat
Hair dryer	heavy	2	-5399.18	hair dryer too heavy
Hair dryer	button	3	-3298.01	complicated button, button not working
Microwave	door	1	-3625.42	door hard to open
Microwave	warranty	2	-3179.48	warranty is not long enough
Microwave	fan	3	-2123.12	fan is loud and inefficient
Pacifier	nipple	1	-1337.39	nipple not in good shape
Pacifier	clean	2	-1053.15	not clean/hard to clean
Pacifier	colors	3	-830.33	colors not the same as advertised

Table 5: 3 lowest-weight words and implied features

3.4 Problem d: Correlation between star ratings and number of reviews

We want to analyze the relationship between changes of number of reviews between arbitrary two months and the average star rating. For each brand, we construct the monthly average rating curve and find the maximum change of number of reviews between two months. We then calculate the average rating before the maximum change and the average rating after the maximum change for each brand. Since the average rating ranges from 1 to 4, we map the average rating to discrete values {1,2,3,4,5} to get better illustration.

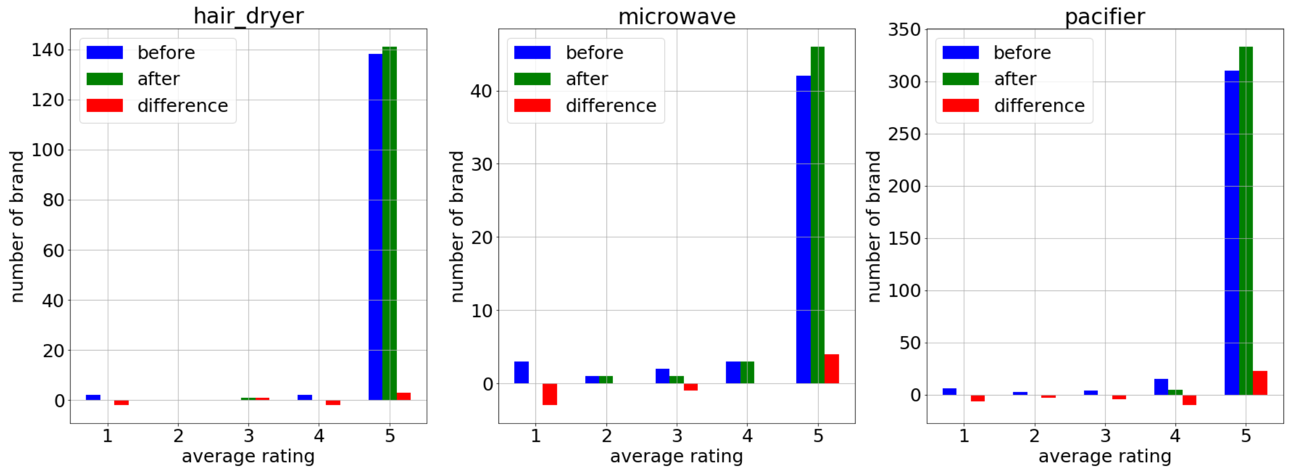

Figure 4: average rating before and after great change in number of reviews by different brands

Figure 4 shows the distribution of different average ratings before and after the maximum change. The red bar shows the difference of rating average before and after the maximum change in each brand. From the plot we can see that most of the red bars are negative, which means that most sudden change of number of reviews tend to result in a higher rating. We can also see that for each type of product most customers tend to have a star rating of 5, which implies that customers are more likely to buy and review products that have high ratings.

3.5 Problem e: Correlation between Quality Descriptors in Text-based Reviews and rating levels

In order to obtain the relation between quality descriptors in text-based reviews and rating levels, for each product, we first group the text-based reviews into five groups according to their star ratings. Then, we extract every single word in the text for all reviews within a same group, sum up the number of times that each word shows up, and the summation will be the frequency (freq) for each specific word. This process is repeated for every product and every group belongs to that product. For convenience, we filter out the stop words such as "I", "he", "on", etc.. The following is the list of the most frequent quality descriptors in text-based reviews corresponding to each rating and product:

Product	**Rating**	**Word, Freq**	**Word, Freq**	**Word, Freq**	**Word, Freq**
Hair dryer	1	('bad', 79)	('old', 73)	('cheap', 72)	('dangerous', 49)
Hair dryer	2	('low', 68)	('old', 65)	('heavy', 63)	('loud', 46)
Hair dryer	3	('heavy', 135)	('fine', 133)	('loud', 72)	('bad', 63)
Hair dryer	4	('good', 950)	('well', 478)	('nice', 305)	('easy', 270)
Hair dryer	5	('good', 1495)	('like', 1366)	('well', 1043)	('best', 966)
Microwave	1	('bad', 53)	('old', 46)	('poor', 34)	('dead', 24)
Microwave	2	('old', 23)	('loud', 17)	('bad', 14)	('damaged', 14)
Microwave	3	('nice', 28)	('fine', 24)	('hard', 17)	('cheap', 15)
Microwave	4	('good', 156)	('well', 116)	('like', 108)	('nice', 72)
Microwave	5	('well', 164)	('small', 162)	('good', 156)	('easy', 152)
Pacifier	1	('old', 148)	('hard', 111)	('big', 74)	('small', 71)
Pacifier	2	('small', 96)	('hard', 87)	('cheap', 47)	('poor', 46)
Pacifier	3	('well', 230)	('better', 168)	('hard', 125)	('clean', 82)
Pacifier	4	('like', 917)	('good', 744)	('easy', 446)	('well', 423)
Pacifier	5	('like', 2135)	('easy', 2135)	('good', 1599)	('well', 1598)

Table 6: List of the most frequent quality descriptors in text reviews corresponding to each rating and product.

From the table, we can easily see that the most frequent quality descriptors heavily depend on the ratings. The descriptors such as 'bad', 'old', 'cheap', 'heavy', 'poor', 'loud', 'hard' are always associated with rating range 1-2. On the other hand, the descriptors such as 'good', 'well', 'nice', 'like', 'easy' are often associated with rating range 4-5. For rating reviews with 3 star, the probabilities to contain negative and positive descriptors are roughly equal. Thus, the negative descriptors are strongly associated with rating range 1-2, and the positive descriptors are strongly associated with rating range 4-5.

3.6 Sales Strategy Analysis

To analyze the target customer, we analyze the words appeared in the reviews and classify the reviews based on the review writer and the purchase time. We then plotted the data into histograms to identify market trend for sales strategy analysis.

We have used the assumption that the number of reviews for a product is roughly proportional to the number of sales, so we can continue the analysis. From Figure 5, it is clear that hair dryers have far more female users compared to male users. So Sunshine company should market the hair dryer toward female customers. It is also noted that young people (<18) and middle-aged people (18–60) are more likely to purchase hair dryers. In addition, more people purchase hair dryers on December, January and February, presumably because people are more like to buy hair dryers in winter.

For microwave ovens, we can see that male customers are much more likely to rate 1 star for the product. Perhaps male customers have relatively higher standard microwave ovens. Sunshine Company probably should focus more on improving the functions that would be appreciated by male customers. Similar to hair dryers, microwave oven's users are primarily young people and middle-aged people. Besides, more people tend to buy microwave ovens during summer and winter.

For pacifiers, the number of male customers are roughly the same as the number of female customers, and most customers are young people. Besides, the number of sales accumulate on January, July, August and December, so Sunshine company can prepare more stock for these months.

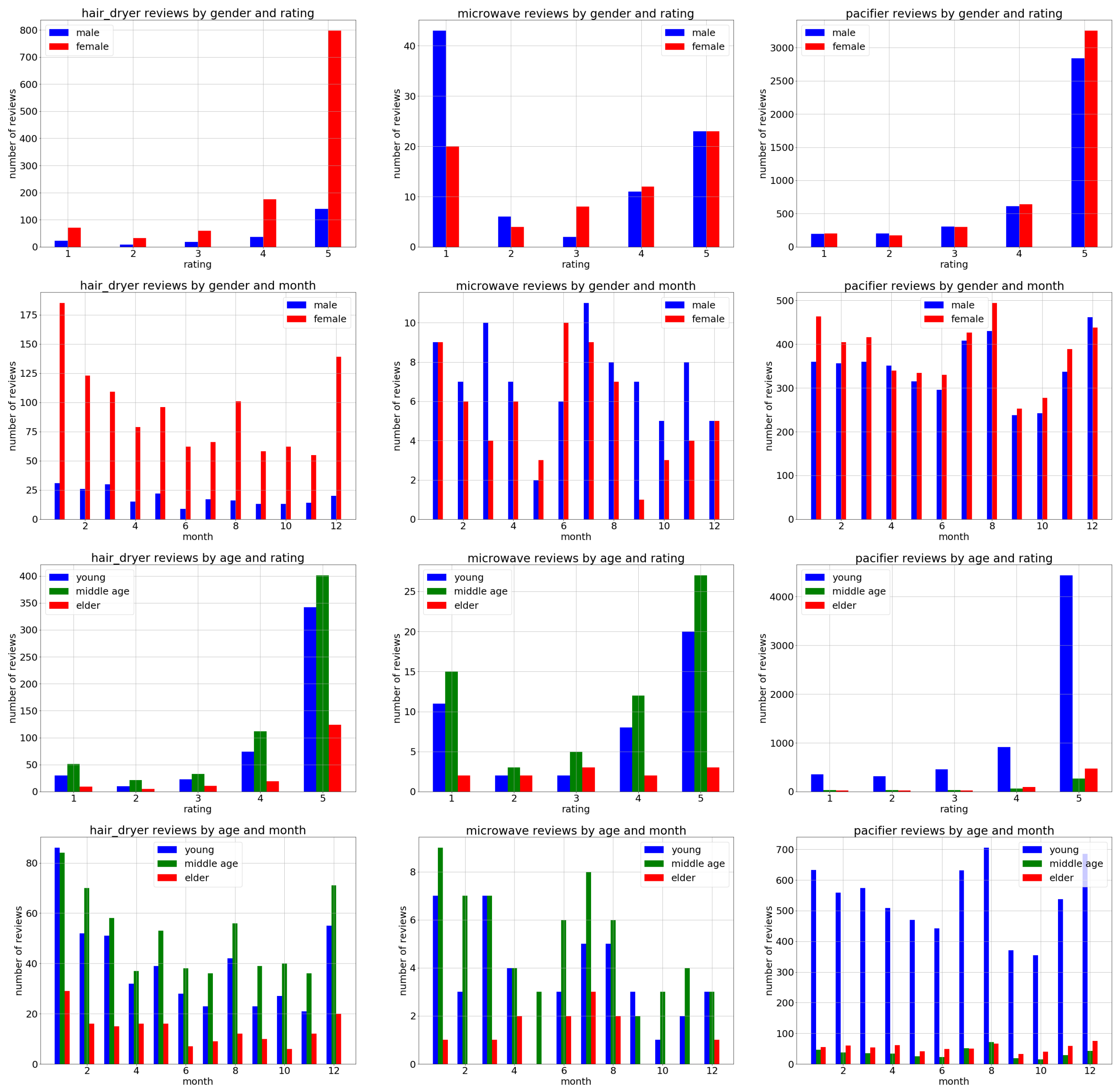

Figure 5: number of reviews categorized by gender and age.

4 Strengths and Weaknesses

4.1 Strengths

We have a standard model of reviews that has been used in three different models. The standard model measures the number of helpful votes, the number of unhelpful votes, and the type of the customer, and it combines all these information into one meaningful score.

The standard model is universal and can be easily applied to measuring different criteria of the reviews, including but not limited to the informativeness and the contribution of the reviews towards the total reputation of a brand. With some modifications, the standard model can be applied to a variety of measurements Sunshine company may need to determine in the future.

Besides, the standard model only contains three parameters, so it isn't an overfit of the dataset. Out of the three parameters, two parameters (γ and δ) have some effect on the score, but the effect is not extreme, since most reviewers are verified buyers and few reviewers are vine customers. Therefore, Sunshine company is free to make any additional adjustment of theses two parameters based on its own proportion of verified and vine reviewers.

We also showed in Section 3.1.1 that our model is stable in terms of the choices of α. We tried three different α's (2.5, e, and 3), the overlapping rate is well above 85%. The stability of the model means that the model provides a true measure for the importance of each review regardless of the choices of parameters. If the model were not stable, different choices of parameters could resulting in different output, and it would be difficult to decided the set of parameters to use and it would be confusing for the company to look at some completely different results.

Our model also measures all the provided information realistically, without a very complicated form. The model follows our assumptions to model each review property using different functions. The functions can be easily coded in a computer program to produce the desired results. If we formulated our model using some differential equations, it would be both analytically and numerically challenging to compute the results.

4.2 Weaknesses and Future work

Our SF Model extracts high ranking words instead of high ranking features. However, the highest ranking words are "like", "good", etc. and the lowest ranking words are "bad", etc. These words are not objective and are not true features of the products. Because of this, some manual labor is needed to read through some high ranking words and manually remove these meaningless words. Even though not much manual labor is required, since there are only a limited number of meaningless words and all duplicates have been combined, we still hope to apply some machine learning technique to remove these words before hand. In this way, the output of SF Model would only contain the features of products.

In addition, since we are only extracting words but not phrases and since one word can have multiple meanings, sometimes one may fail to realize the true meaning of the word in the context of a review. Therefore, it is important to read some reviews containing the specific word to decide what feature the word is representing. But since we are only looking for the highest ranking words, this does not require a large number of work. We hope that if we are given more time, we could formulate a phrase based algorithm to more effectively extract features. We may also use recurrent neural networks to build a natural language processor, and then extract feature using this network. But since training a neural network takes a large amount of time, we are only able to use the word based technique.

Although our model is quite realistic, the dataset we have is not large enough to extract a large set of features. Not many customers provided a detailed, descriptive and objective review. So we are only able to extract a limited number of features that we can provide suggestions for Sunshine company. If we were given a larger set of reviews, we would be able to provide a large number and more detailed suggestions for the company.

5 Conclusion

We have provided several effective models to extract useful information from Amazon reviews of hair dryers, microwave ovens and baby pacifiers. We identified a stable and effective measure (Equation 11) for the informativeness of reviews. We provided a time-base measure (Equation 14) to determine the trend of each brand's reputation with respect to time by observing the reputation vs time plot for each brand. We also formulated a measure (Equation 12) to determine whether a brand is successful or not and extracted both good features and bad features for each product. In addition, we found that customers are more likely to write reviews and rate higher when they saw some negative reviews (Figure 4), and specific words indeed correspond to specific ratings (Table 6). We also found that Sunshine Company should develop special sales strategies based on gender, age and the month of the year. Even though we have provided some strong models, we believe that the models can still be improved. With some future work, we are able to provide more accurate measure.

6 References

[1] Nadeau, R., Cloutier, E., & Guay, J.-H. *New Evidence About the Existence of a Bandwagon Effect in the Opinion Formation Process.* International Political Science Review, 14(2), 203–213. (1993) https://doi.org/10.1177/019251219301400204

*Python Code Used

Listing 1: Code Initialization

```
import pandas as pd
import math
import numpy as np
import nltk
import matplotlib.pyplot as plt
from datetime import datetime
from datetime import timedelta
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk.tokenize import RegexpTokenizer
from nltk.corpus import wordnet as wn
from nltk.corpus import names as nameTypes

nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('names')

end_date = datetime.strptime('09/01/2015', '%m/%d/%Y')
def get_beta(alpha):
  return math.e**(-(1./alpha*math.log(alpha, math.e)))

stop_words = set(stopwords.words('english'))
bad_words = {'aed','oed','eed'}
porter_stemmer = PorterStemmer()
tokenizer = RegexpTokenizer(r'\w+')

df_hair = pd.read_csv('hair_dryer.tsv', sep='\t')
df_microwave = pd.read_csv('microwave.tsv', sep='\t')
df_pacifier = pd.read_csv('pacifier.tsv', sep='\t')
df_hair.replace(np.nan, '', regex=True, inplace=True)
df_microwave.replace(np.nan, '', regex=True, inplace=True)
df_pacifier.replace(np.nan, '', regex=True, inplace=True)

df_all = {'hair_dryer':df_hair, 'microwave':df_microwave,
     'pacifier':df_pacifier}
```

Listing 2: Problem A Code

```
## part a functions
def informitivity(length, num_help, num_unhelp, time, verified, vine,
                  alpha, beta, gamma, delta):
  return math.log(length+1, math.e) * math.log(num_help/time + alpha,
    alpha) * (beta ** (num_unhelp/time)) *
    (gamma ** verified) * (delta ** vine)

def importance(data, alpha, beta, gamma, delta):
  reviews = {}

  for key in data.index:
    length = len(data['review_body'][key])
    num_help = data['helpful_votes'][key]
    num_unhelp = data['total_votes'][key] - num_help
    time = (end_date - datetime.strptime(data['review_date'][key],
        '%m/%d/%Y')).days/30
    if time < 1:
      time = 1
    verified = 1 if data['verified_purchase'][key] is 'Y' else 0
    vine = 1 if data['vine'][key] is 'Y' else 0
    reviews[key] = informitivity(length, num_help, num_unhelp, time,
    verified, vine, alpha, beta, gamma, delta)

  # get the most important 50 reviews by criteria
  return [(k, reviews[k]) for k in sorted(reviews, key=reviews.get,
  reverse=True)]

## part a execution
alpha = math.e
beta = get_beta(alpha)
reviewSave = []
for key in df_all:
  reviews = importance(df_all[key], alpha, beta, 1.1, 2)
  rank = 1
  for product, score in reviews[:3]:
    #print(f"Review of type {key} and rank {rank} has score: {score}")
    #print(df_all[key].iloc[product])
    reviewSave.append([key, product, rank, score] +
        list(df_all[key].iloc[product].values))
    rank += 1
reviewPD = pd.DataFrame(reviewSave, columns =
['dataset', 'product_id', 'rank', 'score']
    + list(df_hair.keys()))
```

```
reviewPD.to_csv('partA.csv')

## part a criteria
alphas = [2.5, math.e, 3]
betas = [get_beta(a) for a in alphas]
total_test = 100

top_same = []
for key in df_all:
  reviews = importance(df_all[key], alpha, beta, 1.1, 2)
  same_set = set()
  for i in range(len(alphas)):
    reviews = importance(df_all[key], alphas[i], betas[i], 1.1, 2)
    curr_set = set([p[0] for p in reviews[:total_test]])
    if len(same_set) is 0:
      same_set = same_set.union(curr_set)
    else:
      same_set = same_set.intersection(curr_set)
  top_same.append(len(same_set)/total_test)

print(f'hair␣dryer␣dataset␣top␣review␣stability:␣{top_same[0]*100}%')
print(f'microwave␣dataset␣top␣review␣stability:␣{top_same[1]*100}%')
print(f'pacifier␣dataset␣top␣review␣stability:␣{top_same[2]*100}%')
```

Listing 3: Problem B Code

```
## part b definition
def reputation(rating, num_help, num_unhelp, time, verified, vine,
                  alpha, beta, gamma, delta):
  return (rating - 3) * math.log(num_help/time + alpha, alpha) * \
         (beta ** (num_unhelp/time)) * (gamma ** verified) * \
         (delta ** vine)

def trend(data, alpha, beta, gamma, delta, diff_weight, dataset):
  # calculate total reputation of all products
  products = {}
  product_start = {}
  product_end = {}
  scores = {}
  length = {}
  total_rep = {}

  # calculate total reputation of all products
  for key in data.index:
    productID = data['product_id'][key]
    if productID not in products:
      products[productID] = 0
      product_end[productID] = key
    product_start[productID] = key
    rating = data['star_rating'][key]
    num_help = data['helpful_votes'][key]
    num_unhelp = data['total_votes'][key] - num_help
    verified = 1 if data['verified_purchase'][key] is 'Y' else 0
    vine = 1 if data['vine'][key] is 'Y' else 0
    time = (end_date - datetime.strptime(data['review_date'][key]
    , '%m/%d/%Y')).days
    score = reputation(rating, num_help, num_unhelp, time, verified,
    vine,alpha, beta, gamma, delta)
    products[productID] += score
    scores[key] = score

  # extract actual date from start and end
  for pid in products:
    product_start[pid] = datetime.strptime(data['review_date']
        [product_start[pid]], '%m/%d/%Y')
    product_end[pid] = datetime.strptime(data['review_date']
        [product_end[pid]], '%m/%d/%Y')
    length[pid] = math.log((end_date - product_start[pid]).days + 1,
        math.e)
```

```
# compute first 30 days average and last 30 days average
#for all products
first30 = {}
last30 = {}
for key in data.index:
  date = datetime.strptime(data['review_date'][key], '%m/%d/%Y')
  pid = data['product_id'][key]
  if pid not in first30:
    first30[pid] = []
  if pid not in last30:
    last30[pid] = []
  if (date - product_start[pid]).days <= 30:
    first30[pid].append(scores[key])
  if (product_end[pid] - date).days <= 30:
    last30[pid].append(scores[key])
for pid in products:
  first30[pid] = sum(first30[pid]) / len(first30[pid])
  last30[pid] = sum(last30[pid]) / len(last30[pid])

# compute final reputation of all products
for pid in products:
  diff_score = (last30[pid] - first30[pid]) / length[pid] * diff_weight
  total_rep[pid] = diff_score + products[pid] / length[pid]

total_products = len(products)
sortedreps = [(k, total_rep[k]) for k in sorted(total_rep,
  key=total_rep.get, reverse=True)]

# plot trend of products with best reputation, worst reputation,
# and some product in the middle
middleRank = 1
bestKey = sortedreps[0][0]
worstKey = sortedreps[-1][0]
middleKey = sortedreps[middleRank][0]
# pacifier worst key doesn't work
if dataset == 'pacifier':
  worstKey = sortedreps[-10][0]

bestTime = []
bestTrend = []
worstTime = []
worstTrend = []
middleTime = []
middleTrend = []
```

```
for key in data.index:
  product_id = data['product_id'][key]
  if product_id != bestKey and product_id != worstKey and
      product_id != middleKey:
    continue
  rating = data['star_rating'][key]
  num_help = data['helpful_votes'][key]
  num_unhelp = data['total_votes'][key] - num_help
  verified = 1 if data['verified_purchase'][key] is 'Y' else 0
  vine = 1 if data['vine'][key] is 'Y' else 0
  score = reputation(rating, num_help, num_unhelp, 1, verified, vine,
                     alpha, beta, gamma, delta)
  if product_id is bestKey:
    bestTime.append(datetime.strptime(data['review_date'][key],
      '%m/%d/%Y'))
    bestTrend.append(score)
  elif product_id is worstKey:
    worstTime.append(datetime.strptime(data['review_date'][key],
      '%m/%d/%Y'))
    worstTrend.append(score)
  else:
    middleTime.append(datetime.strptime(data['review_date'][key],
      '%m/%d/%Y'))
    middleTrend.append(score)

bestTime.reverse()
bestTrend.reverse()
worstTime.reverse()
worstTrend.reverse()
middleTime.reverse()
middleTrend.reverse()

times = []
trends = []
# resample all times to average over 30 days
rate = timedelta(10)
for time, trend in [(bestTime, bestTrend), (worstTime, worstTrend),
  (middleTime, middleTrend)]:
  start = time[0]
  currAvg = []
  new_time = []
  new_trend = []
  for i in range(len(time)):
    currAvg.append(trend[i])
    if time[i] - start > rate:
            new_time.append(time[i])
            new_trend.append(sum(currAvg)/len(currAvg))
            start = time[i]
        times.append(new_time)
        trends.append(new_trend)

    return total_rep, times, trends

## part b execution
i = 0
plt.figure(figsize=(50,75))
plt.rcParams.update({'font.size': 25})
for key in df_all:
    for diff_weight in [0,100]:
        # overall reputation and weighted difference
        _, times, trends = trend(df_all[key], alpha, beta, 1.1, 5,
            diff_weight, key)
        weighted = '' if diff_weight is 0 else '␣weighted'

        plt.subplot2grid((6,3), (i,0))
        plt.title(key + weighted + '␣First␣Rank␣vs␣reputation')
        #plt.xlabel('timeline')
        plt.ylabel('reputation')
        plt.grid()
        plt.plot(times[0], trends[0])
        plt.subplot2grid((6,3), (i,1))
        plt.title(key + weighted + '␣Second␣Rank␣vs␣reputation')
        #plt.xlabel('timeline')
        plt.ylabel('reputation')
        plt.grid()
        plt.plot(times[2], trends[2])
        plt.subplot2grid((6,3), (i,2))
        plt.title(key + weighted + '␣Worst␣Rank␣vs␣reputation')
        #plt.xlabel('timeline')
        plt.ylabel('reputation')
        plt.grid()
        plt.plot(times[1], trends[1])

        i += 1

plt.savefig('partB.png')
plt.close()
```

Listing 4: Problem C Code

```
## part c graph
# target features:
#   type of people
#     categorize by gender
male = set(['he', 'his', 'him', 'husband', 'son', 'father', 'dad',
    'grandpa', 'grandfather', 'boyfriend'])
female = set(['she', 'her', 'wife', 'daughter', 'mother', 'mom',
    'grandma', 'grandmother', 'girlfriend'])
#     categorize by age
young = set(['kid', 'child', 'children', 'son', 'daughter', 'teenager',
    'boy', 'girl', 'teen', 'young'])
youngAge = set(range(19))
middle = set(['husband', 'wife', 'boyfriend', 'girlfriend'])
middleAge = set(range(19, 61))
old = set(['father', 'dad', 'grandpa', 'grandfather', 'mother', 'mom',
    'grandma', 'grandmother'])
oldAge = set(range(61,100))

# histogram by different rating/gender
# total review number by month from different gender
def freqByGender(data):
  total_review = [[0 for _ in range(12)], [0 for _ in range(12)]]
  boys = [0 for _ in range(5)]
  girls = [0 for _ in range(5)]

  for key in data.index:
    line = data['review_headline'][key] + '␣' + data['review_body'][key]
    rating = data['star_rating'][key]
    month = int(data['review_date'][key].split('/')[0])
    words = tokenizer.tokenize(line.lower())
    for word in words:
      stem = porter_stemmer.stem(word)
      if stem in male:
        boys[rating-1] += 1
        total_review[0][month-1] += 1
        break
    for word in words:
      stem = porter_stemmer.stem(word)
      if stem in female:
        girls[rating-1] += 1
        total_review[1][month-1] += 1
        break
  return total_review, boys, girls

# histogram by different rating/age
# total review number by month from different age
def freqByAge(data):
  total_review = [[0 for _ in range(12)], [0 for _ in range(12)],
    [0 for _ in range(12)]]
  low = [0 for _ in range(5)]
  center = [0 for _ in range(5)]
  high = [0 for _ in range(5)]

  for key in data.index:
    line = data['review_headline'][key] + '␣' + data['review_body'][key]
    rating = data['star_rating'][key]
    month = int(data['review_date'][key].split('/')[0])
    words = tokenizer.tokenize(line.lower())
    for word in words:
      stem = porter_stemmer.stem(word)
      if stem in young or stem in youngAge:
        low[rating-1] += 1
        total_review[0][month-1] += 1
        break
    for word in words:
      stem = porter_stemmer.stem(word)
      if stem in middle or stem in middleAge:
        center[rating-1] += 1
        total_review[1][month-1] += 1
        break
    for word in words:
      stem = porter_stemmer.stem(word)
      if stem in old or stem in oldAge:
        high[rating-1] += 1
        total_review[2][month-1] += 1
        break

  return total_review, low, center, high

plt.figure(figsize=(50,50))
plt.rcParams.update({'font.size': 25})
ratings = np.arange(1,6)
months = np.arange(1,13)

# plot classify by gender
counter = 0
for key in df_all:
  total_review, boys, girls = freqByGender(df_all[key])
  plt.subplot2grid((4,3), (0,counter))
  plt.bar(ratings-0.1, boys, width=0.2, color='b', align='center')
  plt.bar(ratings+0.1, girls, width=0.2, color='r', align='center')
  plt.legend(['male', 'female'])
  plt.grid()
  plt.xlabel('rating')
  plt.ylabel('number␣of␣reviews')
  plt.title(key + '␣reviews␣by␣gender␣and␣rating')
  plt.subplot2grid((4,3), (1,counter))
  plt.bar(months-0.1, total_review[0], width=0.2, color='b',
    align='center')
  plt.bar(months+0.1, total_review[1], width=0.2, color='r',
    align='center')
  plt.legend(['male', 'female'])
  plt.grid()
  plt.xlabel('month')
  plt.ylabel('number␣of␣reviews')
  plt.title(key + '␣reviews␣by␣gender␣and␣month')
  counter += 1

counter = 0
for key in df_all:
  total_review, low, center, high = freqByAge(df_all[key])
  plt.subplot2grid((4,3), (2,counter))
  plt.bar(ratings-0.2, low, width=0.2, color='b', align='center')
  plt.bar(ratings, center, width=0.2, color='g', align='center')
  plt.bar(ratings+0.2, high, width=0.2, color='r', align='center')
  plt.legend(['young', 'middle␣age', 'elder'])
  plt.grid()
  plt.xlabel('rating')
  plt.ylabel('number␣of␣reviews')
  plt.title(key + '␣reviews␣by␣age␣and␣rating')
  plt.subplot2grid((4,3), (3,counter))
  plt.bar(months-0.2, total_review[0], width=0.2, color='b',
    align='center')
  plt.bar(months, total_review[1], width=0.2, color='g',
    align='center')
  plt.bar(months+0.2, total_review[2], width=0.2, color='r',
    align='center')
  plt.legend(['young', 'middle␣age', 'elder'])
  plt.grid()
  plt.xlabel('month')
  plt.ylabel('number␣of␣reviews')
  plt.title(key + '␣reviews␣by␣age␣and␣month')
```

```
  counter += 1

plt.savefig('partC.png')
plt.close()

## part c feature extraction function
def wordFreq(data, alpha, beta, gamma, delta, diff_weight, dataset):
  good_words = {}
  bad_words = {}

  reps, _, _ = trend(data, alpha, beta, gamma, delta, diff_weight, dataset)
  sortedKeys = sorted(reps, key=reps.get, reverse=True)
  goodKeys = sortedKeys[:10]
  badKeys = sortedKeys[-10:]

  for key in data.index:
    product_id = data['product_id'][key]
    if product_id not in goodKeys and product_id not in badKeys:
      continue
    line = data['review_headline'][key] + '␣' + data['review_body'][key]
    rating = data['star_rating'][key]
    month = int(data['review_date'][key].split('/')[0])
    words = tokenizer.tokenize(line.lower())
    for word in words:
      if word in stop_words:
        continue
      if product_id in goodKeys:
        if word not in good_words:
          good_words[word] = 0
        good_words[word] += reps[product_id]
      if product_id in badKeys:
        if word not in bad_words:
          bad_words[word] = 0
        bad_words[word] += reps[product_id]

  goodWords = [(k, good_words[k]) for k in sorted(good_words,
    key=good_words.get, reverse=True)]
  badWords = [(k, bad_words[k]) for k in sorted(bad_words,
    key=bad_words.get)]
  return goodWords, badWords

## part c feature extraction execution
for key in df_all:
  goodWords, badWords = wordFreq(df_all[key], alpha, beta, 1.1, 2, 0, key)
  meaningful = goodWords[:100] + badWords[-100:]
  wordPD = pd.DataFrame(meaningful)
  wordPD.to_csv(key+'_wordReps.csv')
for key in df_all:
  goodWords, badWords = wordFreq(df_all[key], alpha, beta, 1.1,
    2, 100, key)
  meaningful = goodWords[:100] + badWords[-100:]
  wordPD = pd.DataFrame(meaningful)
  wordPD.to_csv(key+'_WeightedwordReps.csv')
```

Listing 5: Problem D Code

```
## part d functions
def mapRating(rating):
  if rating <= 1.8:
    return 1
  elif rating <= 2.6:
    return 2
  elif rating <= 3.4:
    return 3
  elif rating <= 4.2:
    return 4
  else:
    return 5

def ratingTrend(data):
  num_review = {}
  timeline = {}
  ratingAvg = {}
  tempTotal = {}
  tempRating = {}

  # get averaged number of reviews in 30-day time intervals
  for key in data.index:
    pid = data['product_id'][key]
    date = datetime.strptime(data['review_date'][key], '%m/%d/%Y')
    rating = data['star_rating'][key]
    if pid not in num_review:
      num_review[pid] = []
      timeline[pid] = []
      timeline[pid].append(date)
    if pid not in tempTotal:
      tempTotal[pid] = 0
    if pid not in ratingAvg:
      ratingAvg[pid] = []
    if pid not in tempRating:
      tempRating[pid] = []
    if (timeline[pid][-1] - date).days > 30:
      num_review[pid].append(tempTotal[pid])
      timeline[pid].append(date)
      ratingAvg[pid].append(sum(tempRating[pid])/len(tempRating[pid]))
      tempTotal[pid] = 0
      tempRating[pid] = []
    tempTotal[pid] += 1
    tempRating[pid].append(rating)
    for pid in num_review:
        num_review[pid].reverse()
        timeline[pid] = timeline[pid][:-1]
        timeline[pid].reverse()
        ratingAvg[pid].reverse()

    # get maximum changes among different products
    posChange = {}
    for pid in num_review:
        if num_review[pid] == None:
            continue
        if len(num_review[pid]) < 5:
            continue
        posIdx = 0
        posMax = -10000
        for i in range(1,len(num_review[pid])):
            diff = num_review[pid][i] - num_review[pid][i-1]
            if diff > posMax:
                posMax = diff
                posIdx = i-1
        posChange[pid] = posIdx

    # get relationship between average rating change and number of
    # review change
    before = [0 for _ in range(5)]
    after = [0 for _ in range(5)]
    for pid in posChange:
        curr_before = sum(ratingAvg[pid][:posChange[pid]+1])
        curr_after = sum(ratingAvg[pid][posChange[pid]+1:])
        beforeIdx = mapRating(curr_before)
        afterIdx = mapRating(curr_after)
        before[beforeIdx-1] += 1
        after[afterIdx-1] += 1

    return np.array(before), np.array(after)

## part d execution
plt.figure(figsize=(30,10))
plt.rcParams.update({'font.size': 25})
counter = 1
ratings = np.arange(1,6)
for key in df_all:
    before, after = ratingTrend(df_all[key])
    plt.subplot('13'+str(counter))
    plt.bar(ratings-0.2, before, width=0.2, color='b', align='center')
    plt.bar(ratings, after, width=0.2, color='g', align='center')
    plt.bar(ratings+0.2, after-before, width=0.2, color='r', align='center')
    plt.legend(['before', 'after', 'difference'])
    plt.grid()
    plt.xlabel('average␣rating')
    plt.ylabel('number␣of␣brand')
    plt.title(key)
    counter += 1
plt.savefig('partD.png')
plt.close()
```

Listing 6: Problem E Code

```
## part e functions
def isAdj(word):
  for tmp in wn.synsets(word):
    if tmp.pos() == 'a':
      return True
  return False

def descriptors(data):
  rateDict = {}
  for i in range(1,6):
    rateDict[i] = dict()

  for key in data.index:
    line = data['review_headline'][key]
    if type(line) is not str:
      continue
    words = tokenizer.tokenize(line.lower())
    rating = data['star_rating'][key]
    for word in words:
      if word in stop_words:
        continue
      if not isAdj(word):
        continue
      if word not in rateDict[rating]:
        rateDict[rating][word] = 0
      rateDict[rating][word] += 1

    line = data['review_body'][key]
    if type(line) is not str:
      continue
    words = tokenizer.tokenize(line.lower())
    rating = data['star_rating'][key]
    for word in words:
      if word in stop_words:
        continue
      if not isAdj(word):
        continue
      if word not in rateDict[rating]:
        rateDict[rating][word] = 0
      rateDict[rating][word] += 1

  return rateDict
```

```
## part e execution
rateDictHair = descriptors(df_hair)
rateDictMicrowave = descriptors(df_microwave)
rateDictPacifier = descriptors(df_pacifier)

freqs = []
alltypes = {'hair_dryer':rateDictHair, 'microwave':rateDictMicrowave,
    'pacifier':rateDictPacifier}
for key in alltypes:
  for rating in alltypes[key]:
    currDict = alltypes[key][rating]
    freq = [(k, currDict[k]) for k in sorted(currDict, key=currDict.get,
        reverse=True)[:50]]
    freqs.append([key]+[rating]+freq)
freqsPD = pd.DataFrame(freqs, columns = ['product', 'rating'] +
    ['wordFreq' for _ in range(50)])
freqsPD.to_csv('partE.csv')
```