Amazon Product Review Analysis

Shouvik Dutta (ID: A20346105) **Harsh Parikh** (ID: A20338453)

Abstract:

Buyers of online shopping sites like Amazon, BestBuy, walmart post reviews of the products that they purchased. These raw reviews provide lot of data and insights of the reviewed product. The number of reviews varies a lot product to product. The general trend is if a product is high seller then such product generally has more reviews. Our project tries to analyze Amazon product reviews and provides an overall ranking as well as custom ranking based on certain product specific features. Lexicon resource based classifier is used to calculate the sentiment score of each collected reviews of a set of products. Top touch-screen specific and popular laptops and some of the popular tablets are considered as dataset for the study. Errors in our analysis and general difficulties regarding the selection of features are analyzed and discussed.

Introduction:

In today's world, the consumer products are moving to the Internet and more and more buyers and sellers are moving to the internet to buy and sell products. The shopping experience has changed in a way that much of the information regarding the products is available online and generated by users. This contrasts, with distant past, in terms of ways the product information used to be spread using techniques like word of mouth and advertising. Since the creation of an online bookstore in 1994, Amazon.com has grown rapidly and been a pioneer for online shopping. Soon Amazon opened its reviews to consumers and eventually allowed any user to post a review for any one of the millions of products on the site. With this increase in anonymous user-generated content, efforts must be made to understand the information that is available in the correct context and develop methods to determine the intent of the reviewer. Understanding what online users think of its content can help a company to market its product as well as maintain its online reputation.

The purpose of this project is to investigate a small part of this large problem i.e. overall positive and negative attitudes towards a set of products and also checking the feature specific positive and negative attitudes towards those products. Sentiment analysis attempts to determine which features of text are indicative of its context (positive, negative, neutral) and build a system that takes advantage of these features. The problem of classifying text as positive or negative is not the whole problem but it does offer a simple enough premise to build a ranking system that makes use of it.

Amazon employs a 1-5 scale for all its products regardless of their category, and this leaves a void to determine the advantage and disadvantages to different parts of a product. Problem with such rating system is a potential buyer is not able to rank the products based on a specific requirement.

Data:

Although Amazon does not have an API like Twitter which directly provides all the reviews, it does have Amazon Advertising API which provides review related data in form of IFrames. We make use of an

Amazon Scraper [i] created by Adam Griffiths for extracting reviews from the IFrame which is an html page. For our project, we are collecting reviews for 40 products of laptop and tablet types. We store all Product ID, Title, Manufacturer in a single text file named 'items.txt'. We store Review ID and Review text in files which are named after the product ID. E.g. Product1.txt would have all reviews for Product1. We have created a dictionary named 'product_dictionary' that stores product ID as key and (title, manufacturer) tuple as value. We create another dictionary name 'product_reviews' that stores product ID as key and a list of tuples, each tuple having a review for that product and its sentiment score which is calculated using sentiwordnet senti_synsets method. A more specific case of the issue is the case of negation. Words that occur after a negated word in an article have opposite meanings than what they originally meant and present noise in the data if they are saved as unigrams. For example, the phrase "not like" would indicate a negative review, while "not" and "like" independently would not necessarily give the same classification. This problem was addressed by adding a not to the word which follows a negation word while tokenizing. So if we encounter "not like" then the tokenizer function will tokenize as "not" and "not_like".

We were able to capture 1322 positive reviews and 1424 negative reviews from the whole set of products that was considered for analysis.

Methods

After collecting reviews, we are tokenizing each review and we considering all the cases like handling negation, emoticons, URL, etc. Then we are finding the average sentiment score for each product and feature based score.

Feature Vector Representation:-

In this project, the features are modeled as a set of words i.e. the documents are represented as unordered collections of words without giving any consideration for grammar. The training data D consists of a corpus of n documents (reviews) where xi for i=1, … , n represents the collection of reviews and |V| represents the dictionary containing m distinct words. In this model each word wj for j = 1,…,m is represented as a feature. Each word is assigned an integer id. The number of times word wj occurs in document i is stored in X[i,j] where j is the value of the feature. X is an n-by-m sparse matrix that only stores the non-zero portions of the feature vectors.

Classification: -

Naive Bayes Classification[ii]: - Naive Bayes is a simple but robust classifier applied using Bayes' Rules that yields very useful results. It assumes the independence of each of the features in the vector and for each feature, calculates the probability that it will appear given the class. The probability of a class given the feature set is simply the product of the probability that the class will occur and the probabilities of each of the feature vectors. This process is repeated for each of the possible classes and the text is classified according to the maximum probability. The formal definition is given as:

$$\hat{s} = s\epsilon S argmax P(s) \prod_{j=1}^{n} P(f_j|s)$$

where

$$P(s_i) = \frac{count(s_i, w_j)}{count(w_j)}$$

$$P(f_j|s) = \frac{count(f_j, s)}{count(s)}$$

Decision Tree Classification[iii]: - The Decision list a rule-based tagger that has the advantage of being human readable. One of the major problems with Naive Bayes is the difficulty in identifying which probabilities are causing certain classifications. One can look at the errors made, but there's no way to actually know if there is some feature that is at the root of the problem. The format of a decision list alleviates this problem by making the classification rule-based. The classifier must only determine the existence of the feature at each level and tag appropriately if the feature is in the document. This makes is very easy to intensify which rules the classifier thinks are important, and aids in the removal of features that are causing inaccurate rules.

Experiments:

Sentiment Score Calculation

In this project, we calculate Sentiment Score using SENTIWORDNET[iv] which is an Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. SentiWordNet assigns to each synset of WordNet in three type of Sentiment scores: positivity, negativity, neutrality.

So for every review in our project, we calculate sentiment score and make a dictionary of all the reviews. In this project, first we have calculated overall sentiment score per product and then we have made a histogram of it.

Item Number	Product Title	Sentiment Score
1	Microsoft Surface 2 (32 GB)	1.94442
2	Dell Inspiron i7359-6790SLV 2-in-1 Touchscreen Laptop	1.74788
3	G-Tab Iota Quad Core Android Tablet PC	1.45588
4	2015 Newest Dell Chromebook, 11.6 inch HD Non-Touch LCD	1.25
5	Microsoft Surface Pro 4, Intel Core i5	1.23724
6	Dell Inspiron 3000 15.6-Inches Windows 8.1 Laptop	1.05556
7	Microsoft Surface 3 Tablet	0.94
8	Dell 15 5000 i15547-5003sLV 16-Inch Touch Screen Laptop	0.923913

Fig 1. Overall Ranking of Few products.

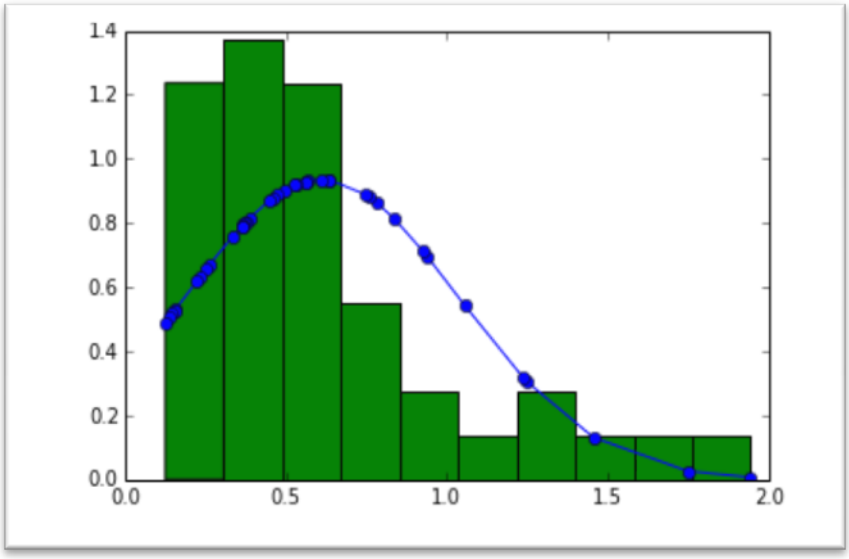


Fig 2. Histogram of Product Average

Moreover, for feature based ranking, we are sorting products based on its features (i.e touchscreen or processor) rank. Sometimes a buyer only wants good touchscreen laptop and he is not bothered about overall ranking then we provide this sort of facility as well.

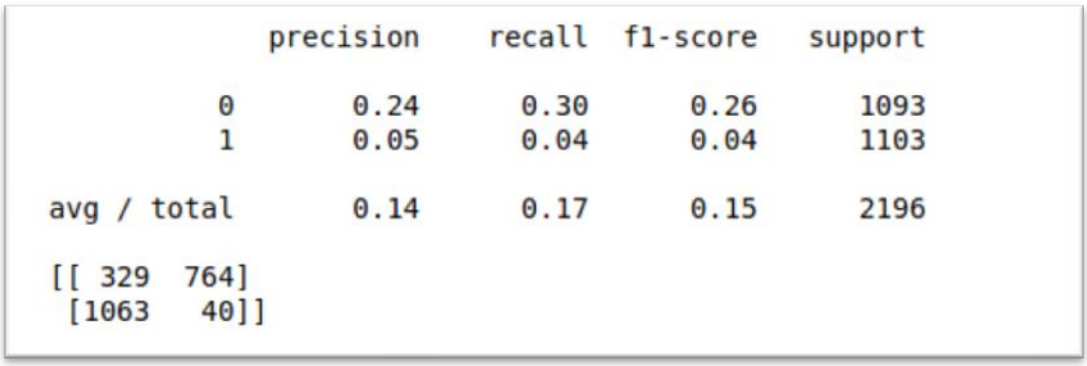
Item Number	Product Title	Sentiment Score
1	Microsoft Surface 2 (32 GB)	5.29421
2	Acer Aspire Switch 10 E SW3-013-1566 2-in-1 Tablet & Laptop	4.25
3	Dell Inspiron i7359-6790SLV 2-in-1 Touchscreen Laptop	3.40217
4	Dell Inspiron i3000-5099SLV Touchscreen Laptop	3.375
5	Microsoft Surface Pro 4, Intel Core i5	2.41667
6	Dell Inspiron 15 i5548-4167SLV - 15.6" Touchscreen	2.33333
7	G-Tab Iota Quad Core Android Tablet PC	2.2
8	Dell Inspiron 15 7000 Series i7537T Touchscreen Laptop	1.96685

Fig 3. Feature (touchscreen) Based ranking of few products

By comparing fig 1 and fig 3, we can see that the product which has higher feature score doesn't have a good overall rating. For an example, Item 2(Acer Laptop) in fig 3 has good touchscreen feature but it does not fall in best overall laptop (fig 1) category.

So this way, we have done feature based ranking by its processor, touchscreen and storage.

Now, let's see how our classifiers are working. So here's our table summarizing our results.



	precision	recall	f1-score	support
0	0.24	0.30	0.26	1093
1	0.05	0.04	0.04	1103
avg / total	0.14	0.17	0.15	2196

```
[[ 329  764]
 [1063   40]]
```

Fig 4. Result

We have used Naïve Bayes Classifier and Decision Tree Classifier. We are getting maximum accuracy by using our best settings (i.e min_df = 3 and max_df = 0.4).

Conclusion:

Generally, the results of this project were quite successful. The classifier managed to accurately tag a good amount of user-generated reviews with an accuracy of 67% in naïve bayes classifier and 80.33% in decision tree classifier. We did observe that there were some outlier products in our project as those products had very few reviews (less than 50). Such product scatters our ranking and makes our ranking graph with long tails on positive side. This is due to a general human nature of putting positive reviews more than negative reviews. So we assume if a product has fewer reviews the odds are it would have more positive reviews than negative ones leading to high sentiment score. We observed such a tendency in a product which has only 6 reviews but still managed to get ranked on the higher side of our overall ranking histogram.

This project could be further extended to provide more feature based ranking based on the analysis of the users review. Also the basic analysis of reviews does provide a general trend on the preference of users and manufacturing companies can analyze this pattern and improve their product based on the users reviews. As for e.g. suppose Product A has a fair good rating and is ranked in top 10 products in overall ranking. But when we look at product A ranking based on a specific feature it might be ranked much lower. This insight observed by user review pattern helps the manufacturer to improve on that specific feature for which his/her product is ranked poorly leading to increasing the products overall ranking.

References:

i. Amazon Scraper by Adam Griffiths (https://github.com/adamlwgriffiths/amazon_scraper)
ii. Naïve Bayes Classifier (http://scikit-learn.org/stable/modules/naive_bayes.html)
iii. Decision Tree Classifier (http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html)
iv. SentiWordNet (http://lrec.elra.info/proceedings/lrec2010/pdf/769_Paper.pdf