

Ranking products from online customer reviews

Anushka Gupta
anushka21114@iiitd.ac.in
MTech,IIIT Delhi

Nitin Khatkar
nitin21133@iiitd.ac.in
MTech,IIIT Delhi

Shoumik Bhattacharya
shoumik21144@iiitd.ac.in
MTech,IIIT Delhi

1 MOTIVATION AND PRECISE PROBLEM STATEMENT

Rank the entered products while comparing aspect-based rating of the particular product using reviews from e-commerce websites. Our task is to rate entered product on various aspect of that product, let's say input be one mobile phone onto our software, then it would be able to rate that mobile phone on basis of various aspects such as battery life, screen size and value for money among other aspects while fetching reviews from multiple e-commerce websites. Then, for same product it will give feedback to user for various aspect-based ranking for the product and suggesting user the website with best price. Further, using aspect-based ranking of different products of same category it would rank product and then, suggest user with the most appropriate product. With numerous products launching in almost all e-commercial giants every now and then, and user with limited amount of time in hand, nobody likes to visit each and every website and then, read reviews about it and then again repeat the same task for other product and then, decide which product to buy (Refer Figure 1), wouldn't it be easier if there existed a website which could do all this for you. To make this process hassle free, what we intend to solve these problems and provide user with one stop solution where user can enter products which are in his/her mind, and then we would be able to rank those products while comparing their aspect-based rating on various parameters related to the product.

2 LITERATURE REVIEW

Literature review of some of the existing work in the domain of our problem.

- Gobi et. al.[4] used customer reviews of mobile phones and cameras from Amazon dataset. They applied a four step approach which included Pre-processing the dataset, Feature extraction, Context Identification and Fuzzy Ranking. The Pre-processing of data included Tokenisation and PoS Tagging. The Feature Extraction step included extracting the Explicit and Implicit Features such as Battery life, price, camera and Size, Weight etc. respectively. They took all synonyms of explicit features and clubbed them under a common category. For implicit features they took the count of each feature and the most frequently occurring one was assigned as implicit feature to the opinion word. The adjectives in the pre-processed reviews represent opinion words. In the Context Identification step Linguistic rules were used to handle opinion words. For example, "the battery life is very long". It is not clear whether "long" means a positive or a negative opinion on the product feature "battery life". "This camera takes great pictures, but has a short battery life." Here the word "but" changes the opinion of the sentence. "Great" represents a positive opinion and "short" has a negative opinion.



Figure 1: How user need to search different websites and compare products manually

Then they applied Fuzzy Topsis Ranking in which the opinion words were classified into five categories such as very good, good, medium, poor and very poor based on the semantic similarity between them. Then Five triangular fuzzy numbers are used for scoring the features. They obtained Precision, Recall and F-measure values of 0.795, 0.562 and 0.658 respectively and provided an efficient ranking system based on the customer reviews.

- Karthik R.V. et. al.[5] devised a new method algorithm called Feature Based Product Ranking and Recommendation Algorithm(FBPRRA) for recommending products to the users. They wanted to make a generalized sentimental analysis for the same for handling of all varied type of input, unstructured data from review blogs and social networking sites with customer specific recommendation. Occasion(anniversary, birthday) and actual reviewer information(mainly age) is identified from the review blogs or reviews given, then according to user inputted occasion and user age, they identified the correct product for the user entered occasion and age for the end user. Then user interested features and weightage calculation is automatic. Took online reviews and target user age as input, then identified correct reviewers from the lot and calculated weightage(that is, positive, negative or neutral)

and then according to occasion they filtered them further and gave recommendation on basis of that. Processing of reviews is in the order of tokenizer, filter tokens, filter stop words, transform cases and then applying FBPRRA algorithm to it. In the proposed algorithm they got a accuracy of about 81.23 percent which was higher than the other learning techniques namely SVM, Decision Tree, Naive Bayes, KNN and Random Forest. Further they applied the same algorithm to amazon dataset taken and found out that out of total data for given products nearly only 30 percent filtered out to be useful for applying information retrieval according to given target user for a given occasion. Hence, on basis of end user's information mainly age and other factors like occasion, they calculated weight for each feature and create a ranking and recommendation system.

- Ping et. al.[2] reviewed two types of sentiment analysis: Lexicon-based sentiment analysis and Sentiment analysis based on machine learning. They found that a combination of both techniques could be found in existing literature. They stated to make effective decisions refined accurate analysis is required. They proposed information fusion to add some information from consumers for which ranking is to be done using sentiment analysis. The approach of using information fusion in ranking products is based on intuitionistic fuzzy theory, use of weighted directed graphs and hesitant fuzzy theory, providing viable and effective product rating algorithms. The approach they proposed performed accurate analysis on a large number of product reviews from the internet.
- Lokhande et. al. [3] proposed a real time search engine where users can find the best product after comparing multiple e-commerce websites like Flipkart, Amazon and Snapdeal. The system compares the price of the product among different websites and then ranks them in descending order. They have 2 possible cases- One, if the product is searched for the first time. This case requires a lot of computational power. The second case is that if the product already exists in the database. In such a case, only the product prices are updated every hour by web crawlers which is a very quick process. The Product ranking algorithm ranks the products based on popularity, prices, ratings and review counts. Sentiment analysis is performed on the user reviews using Word Cloud. Naive Bayes was used for Text classification. First the user entered the product which needs to be searched, then Web crawler crawls web pages based on certain parameters and filtered out the unnecessary products, selected products are stored in a csv file and in the database, then products are ranked in decreasing order based on the ranking algorithm which considers all the parameters and then displayed to the user.
- T. Saranya et. al.[7] identified that a large number of customers have shifted to the online mode of shopping from various E-commerce websites like Amazon, Flipkart, Myntra, etc because they are more convenient, require less time and are cost effective. As a result, customers post product reviews in the E-commerce websites, expressing their reviews and opinions about the product. But, at the same time, it has

become very difficult for the customers to purchase products based on their pictures and short descriptions, since they can't judge the quality and performance of the product by merely looking at its image. To tackle this issue, one solution is to rank each product based on the features obtained from the customer reviews. To implement this, they proposed a method "feature-based opinion summarization of reviews" which involves mining of the product features, on which customers have given their opinions and then ranking them based on their frequency of appearance in the reviews. The paper focuses on product reviews on various websites like amazon, etc. They collected the reviews of a product and loaded into the database, which is then fed to the product ranking system. First they extracted product feature followed by categorizing them into implicit and explicit features. Then, they pruned irrelevant features and labelled sentence involving subjective and comparative sentences followed by assigning sentiment orientation to the sentences. Then, they used Ranking algorithm 'Prank' to rank the products. Finally they calculated performance of the ranking system is evaluated using Precision and Recall. They ranked system to produce better results comparing to previously existing system, where the products were ranked based on their overall quality and the features mentioned by the user.

- Syamala et al.[6] used an amazon product URL to get all the reviews on that product and then these reviews are preprocessed. They calculated the polarity of each word and then calculated the sum of polarities. This total words polarity score was then normalized using the formula $\text{Normalized Polarity} = \text{Polarity} \cdot \text{MIN}(\text{Product-rating}) \cdot (\text{MAX-MIN})$, where Product rating is the rating on that review out of 5. Next they used Principal Component Analysis for feature selection. This normalized polarity score and a vector of features was then fed to Ada boost and Random Forest Classifiers for Prediction and Classification of results.
- Bhatt et al.[1] took reviews of iPhone 5s on Amazon and applied four different types of processing. The first method was based on ratings on the review. They classified the review score as 1 if the rating on the review was 4 or 5, the review score was 0 if the rating was 3 and the review score was -1 if the rating was 1 or 2. In second methodology they calculated a polarity score of the entire review. In third methodology, they extracted all the words in the review after the word "NOT" and then preprocessed the string and calculated the polarity score of the words after "Not". Then they inverted it since those words are prefixed by a "Not". In the fourth methodology, they preprocessed the whole review by removing the stopwords, converting to lowercase, removing numerics, whitespaces and special characters and then calculated the sum of polarity scores of all words in the review and classified them. They gave equal weightage to all the four types and then calculated the overall polarity of the review and classified them as positive, negative or neutral.

3 BASELINE RESULTS

First Baseline Model

Methodology -

- (1) Create a text file consisting of different mobile phone Urls on amazon. Read this text file and extract data from amazon on these urls.
- (2) Create a dataframe consisting of product title, product price, overall rating, total reviews, availability.
- (3) Calculate product popularity based on answered questions on the product and append a column for popularity on the dataframe.
- (4) Clean the data where there is NA present in the dataframe and drop those rows.
- (5) Extract the ratings and total review counts as float values and append to the dataframe.
- (6) Now we will normalize the values of four columns- popularity, ratings, price and total review count.
- (7) After normalization, give equal weightage to all four attributes and rank the products. Here, we will subtract the value of price, because the lower the price the better the product, where the higher the number of reviews, ratings and popularity, the better is the product.
- (8) Create a sorted dataframe based on the above point and this dataframe is used for ranking the products.
- (9) Extract top 5 products from the ranked list of products.
- (10) Use web scraping to Get all reviews on the top 5 products. Create a dataframe consisting of customer name, review title, rating and review text.
- (11) Extract only the review text from the above dataframe and plot a wordcloud consisting of reviews of each product in top 5.
- (12) Next, we will do sentiment analysis on reviews using Naive Bayes.
- (13) On each review of each top 5 products, get the polarity score and save in a column in the dataframe.
- (14) Preprocess each review. Convert to lowercase, Remove stopwords, numerics, special characters and whitespaces and store the preprocessed string in a column in a dataframe.
- (15) Based on the polarity score, classify the review as positive if polarity>0, negative if polarity<0 and neutral if polarity=0.
- (16) Split the dataset into training and testing with feature matrix as preprocessed query and target matrix as polarity score.
- (17) Vectorize the query and train the Multinomial Naive Bayes model.
- (18) Apply the model on the test data for predicting the sentiment.
- (19) Get the total number of positive and negative review counts for each product.
- (20) Calculate the sentiment score by dividing total number of positive and negative reviews with total reviews and giving a score of 1 and -1 respectively.
- (21) Give equal weightage to buying probability based on raking and the sentiment score of reviews and get Total score.
- (22) Sort the products based on final score.

Result - Naive bayes classification score 90 percent.

Second Baseline Model

Methodology -

- (1) Extracted all reviews on Redmi 9 on Amazon and created a dataframe consisting of Customer Name, Review Title, Rating and Review text.
- (2) Extracted the Ratings as a float value and appended a separate column in the dataframe.
- (3) **Method 1** - Give a score of 1 to the review if the rating on a review is 4 or 5, give a score of 0 if the rating on a review is 3, and give a score of -1 if the rating is 0 or 1. Append this score in a new column in the dataframe.
- (4) **Method 2** - Generate a polarity of each review text. If the polarity>0, give a score of 1, if the polarity<0, give a score -1, and if 0, give a score of 0. Append this score as a new column in the dataframe.
- (5) **Method 3** - Extract all the reviews which have the word "Not" in them. Now get all the words in the review after the word "Not". Preprocess this string after the "Not" by converting to lowercase, removing numerics, special characters and stopwords and whitespaces. Now get the polarity score of each word and sum it. If the sum >0, give a score of 1, else -1 and if the sum is 0, then 0. Now invert these signs because this string is prefixed with a "Not" so it has the opposite sentiment. Append this score in a new column in the dataframe.
- (6) **Method 4** - Preprocess the whole review text string by converting to lowercase, removing stopwords, numerics, special characters and whitespaces. Now calculate the polarity of this preprocessed words and get the sum of polarities of the words. If the sum>0, give a score of 1, if sum<0 give a score -1 and if the sum is 0, give a score of 0. Append this score in a new column in the dataframe.
- (7) Give equal weightage to all the above 4 methodologies and get the average score of each review. If the average score is positive, the review is positive, else negative, and neutral if the average score is 0.
- (8) Get the total number of positive and negative reviews on a product and advice the user based on the positive and negative count.

Result - based on the positive and negative review count we advice the user to either buy or not buy the product. Refer to Figure 2

Third Baseline Model

Methodology -

- (1) Get all reviews on Redmi 9 Active on Amazon and create a dataframe consisting of Customer Name, Review Title, Rating and Review text.
- (2) Extracted the Ratings as a float value and appended a separate column in the dataframe.
- (3) Get the polarity of each review and if polarity>0 give a score 1, if polarity<0 give a score -1, and 0 if polarity is 0.

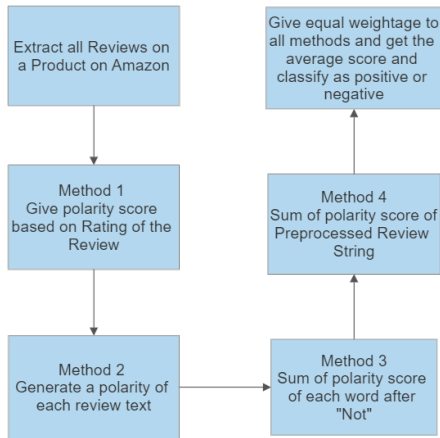


Figure 2: Workflow of the Second baseline model

- (4) Normalize the polarity values using the formula- $\text{Normalized Polarity} = \text{Polarity} \cdot \frac{\text{MIN}(\text{Product-rating})}{(\text{MAX}-\text{MIN})}$
- (5) Append the Normalized Polarity values in a separate column in the dataframe.
- (6) Apply preprocessing on all reviews by converting to lower-case, removing stopwords, numerics, special characters and whitespaces. Append this preprocessed review string in a separate column in the dataframe.
- (7) Split into training and testing data with feature matrix as preprocessed reviews and target matrix as polarity values.
- (8) Vectorize the review text to numbers.
- (9) Apply Principal Component Analysis to reduce the number of features.
- (10) Trained Adaboost classifier model and Random Forest classifier and predicted the reviews as positive or negative. Predicted on the test data to get the results.
- (11) Plotted a decision tree based on Gini index value.

Result - Random Forest Classifier gave a score of 87.5 percent.
Refer to Figure 3

4 PROPOSED METHOD

We intend to solve our identified problem in following phased manner. Refer to Figure 4

- (1) Scraping of reviews from different e-commerce giants, like Amazon and Flipkart, with features like rating, availability, price, and reviews.
- (2) From each set of reviews, we do aspect based rating on the basis of certain fixed features such as Design, Display, Software, Performance, Battery Life, Camera and Value for Money.
- (3) Calculate polarity of sentiment for the above mentioned features while training models such as Naive Bayes on sample set of reviews and testing on rest of the reviews and assigning score on the basis of polarity value obtained.

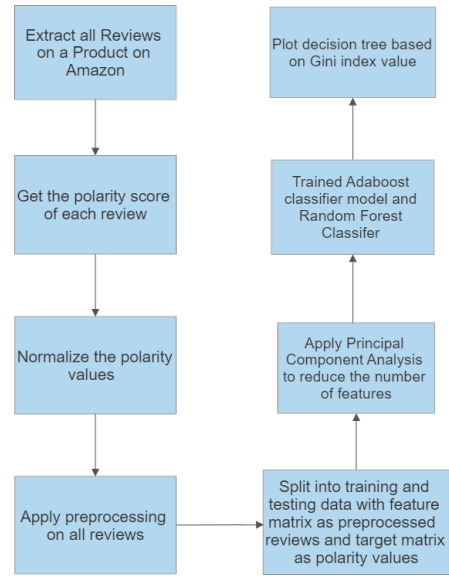


Figure 3: Workflow of the Third baseline model

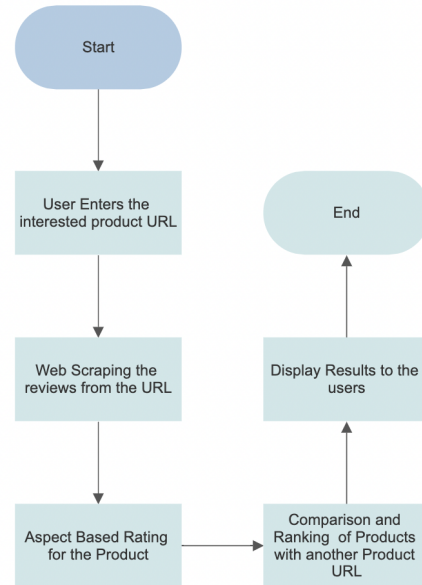


Figure 4: Planned Workflow for the project

- (4) Using above all mentioned features, we calculate the average score for the device entered by giving appropriate weight to each of the mentioned features.
- (5) For all the entered list of devices, we calculate the individual feature score as well as the overall score for the device, and use it for ranking of devices among each other.
- (6) Finally, we will design a website for implementing the stated algorithm.

REFERENCES

- [1] Aashutosh Bhatt, Ankit Patel, and Kiran Gawande Harsh Chheda. "Amazon Review Classification and Sentiment Analysis". In: *International Journal of Computer Science and Information Technologies* 6 (2015), pp. 5107–5110.
- [2] Zhi-Ping Fan, Guang-Ming Li, and Yang Liu. "Processes and methods of information fusion for ranking products based on online reviews: An overview". In: *Information Fusion* 60 (Feb. 2020). doi: 10.1016/j.inffus.2020.02.007.
- [3] Suyash Lokhande et al. "Product Rank Algorithm Along with Sentiment Analysis on Reviews of E- Commerce Websites". In: 2021.
- [4] Gobi Natesan and A. Rathinavelu. "Analyzing cloud based reviews for product ranking using feature based clustering algorithm". In: *Cluster Computing* 22 (May 2019). doi: 10.1007/s10586-018-1996-3.
- [5] Karthik R V, Ganapathy Sannasi, and Kannan Arputharaj. "A Recommendation System for Online Purchase Using Feature and Product Ranking". In: Aug. 2018, pp. 1–6. doi: 10.1109/IC3.2018.8530573.
- [6] Maganti Syamala and Nattanmai Jegathan Nalini. "A Filter Based Improved Decision Tree Sentiment Classification Model for RealTime Amazon Product Review Data". In: *International Journal of Intelligent Engineering and Systems* 13 (Feb. 2020), pp. 191–202.
- [7] T.Saranya. "Mining Features and Ranking Products From Online Customer Reviews". In: *International Journal of Engineering Research Technology* 2 (Oct. 2013), pp. 643–648.