

University of Connecticut  
School of Business

***OPIM 5894***

***Data Science with Python***

**Consumer Reviews of Amazon Products**

**Team 03**

**Janani Vasudevan**

**Silky shah**

**Vinay Vihari Lakamsani**

**Radhika Malviya**

**Xiaoxu Xing**

INDEX

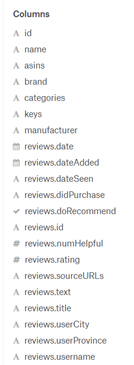
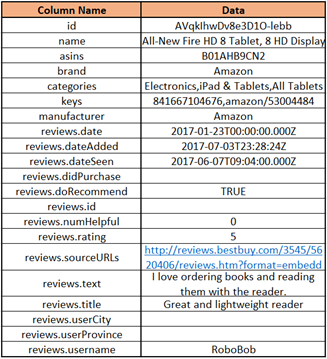
* Data Description
* Data Pre-Processing
* Exploratory Data Analysis
* Modelling
* Findings
* Deep dive for a product
* Recommendations
* Code Links

**Consumer Reviews of Amazon Products**

**DATA DESCRIPTION:**

**Data source**

The data has been extracted from the Kaggle website<https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products>. The dataset includes basic product information, rating, review text, and more for each product. It contains about 37 attributes and a list of over 34,000 consumer reviews for Amazon products. The customer reviews are included which has both rating and text. The columns in the data file and an example record are as shown below..

**Data variables**

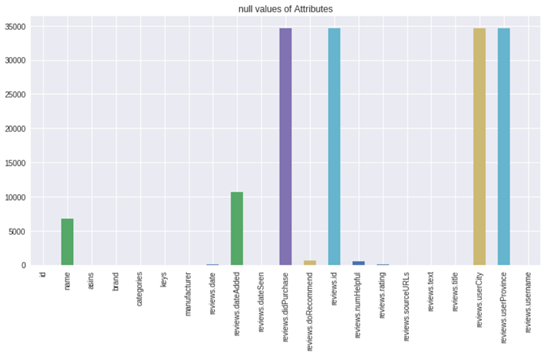
The columns have various data types such as dates, numerical and categorical variables as seen above. The basic product information has been explained by the columns name, asins(Amazon Standard Identification Number), brand, categories, keys, manufacturer. The column id is used to identify users that have provided the respective reviews. The columns that provide us details regarding reviews are the most informative columns. Among the review columns, we consider the most significant columns as reviews.date, reviews.dateAdded, reviews.text, reviews.title.

**Missing values and unique values analysis**

The attributes sourceURLs, text, title, username have the maximum unique values.

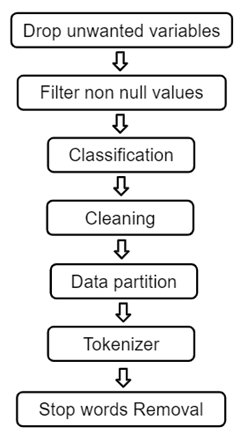
The missing value analysis is indicated in the below figure and plot. The attributes didpurchase, reviews.id, userCity, userProvince have the maximum missing values in the range of 34000. Considering that the entire dataset itself is 34700 in size, these columns provide little insight and can be eliminated from further analysis.





**DATA PRE-PROCESSING:**

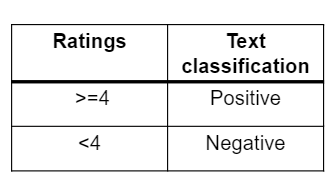
The process flow of data preprocessing is as shown below. We have chosen the steps in order to aid the goal of sentiment analysis of the reviews.



· **Variables chosen** are Date, Date added, Rating, Review text, Review title as these form to be the most insightful attributes for the case of sentiment analysis of reviews.

· **Filtering non null values** as explained in the missing values sectionand are assigned to the array “check”

· **Classifying** reviews as positive and negative based on the ratings provided. The reviews with ratings greater than or equal to 4 are classified as positive. The reviews with ratings lesser than 4 are classified as negative.



· **Cleaning** **the review text**

• Converting to string

• Convert text to lower case

• Removing characters like symbols

· **Data partition** into Training and test.

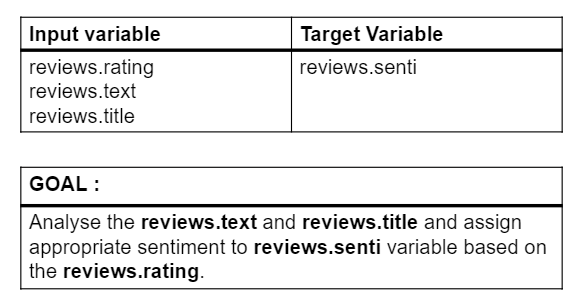
•Train: 75%

• Test: 25%

· **Tokenizing** each review text into words, for both training and test data set

· **Stop Words** such as articles and prepositions are eliminated

To summarize the data description and preprocessing, we use the initiated variable reviews.senti as the target variable as shown below.

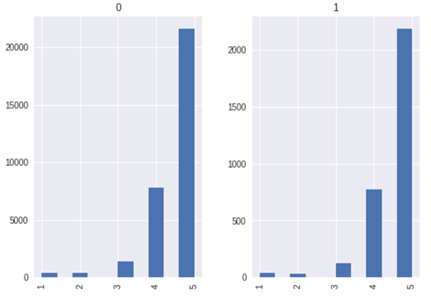


**EXPLORATORY DATA ANALYSIS:**

We tried to uncover some trends by identifying some interesting facts about the data:

1. Only 0.55% of users are bulk users
2. Around 9% of ratings have been submitted by 0.55% of users

Below is the graph of Rating vs Users Graph where 0 = Non-bulk users and 1 = Bulk users



We also checked distribution of User rating vs Users:



We also checked Net promoter score of Amazon by using the below formula:

•Rating 1,2,3 - Detractors

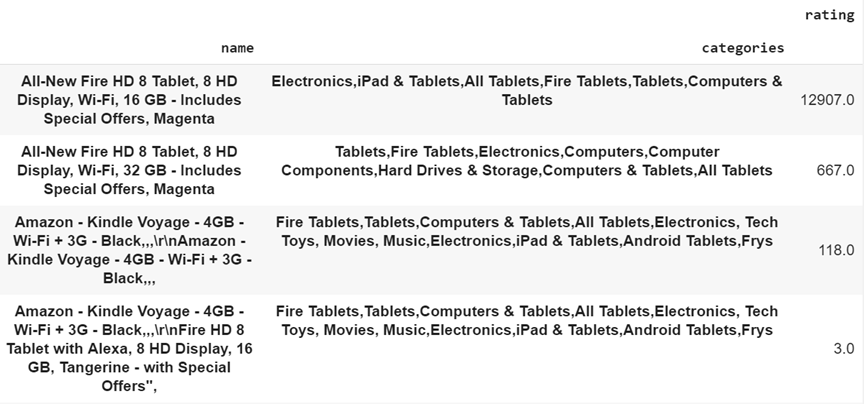
•Rating 4 - Passive

•Rating 5 - Promoters

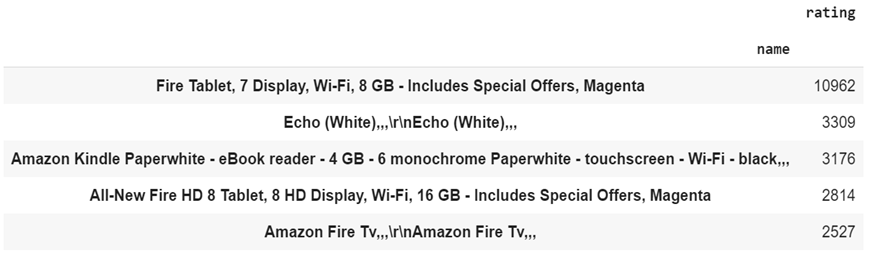
•NPS = (Promoters - Detractors)/Total ratings \* 100

We found that NPS Score of Amazon(overall) is 61.99.

Later, we identified most reviewed product on Amazon in last 90 days and found that All-New Fire HD 8 Tablet and Amazon - Kindle Voyage were most popular. Possible reason maybe that there might be some offers running on these products or the product is newly launched.



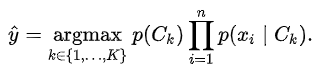
We also checked most reviewed product on Amazon. In our dataset, below are the most reviewed products on Amazon for the date timeframe data we had:



**Modelling:**

**Naïve Bayes Classifier**

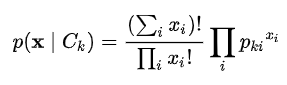
In [machine learning](https://en.wikipedia.org/wiki/Machine_learning), naive Bayes classifiers are a family of simple "[probabilistic classifiers](https://en.wikipedia.org/wiki/Probabilistic_classifier)" based on applying [Bayes' theorem](https://en.wikipedia.org/wiki/Bayes%27_theorem) with strong (naive) [independence](https://en.wikipedia.org/wiki/Statistical_independence) assumptions between the features.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. 

Library Used: from nltk.classify import NaiveBayesClassifier

**Multinomial Naïve Bayes:**

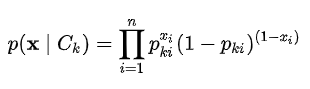
Models the number of discrete counts of feature. With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by a multinomial (p1 …., pn) where pi is the probability that event I occurs (or K such multinomial in the multiclass case). A feature vector x= (x1…, xn) is then a histogram, with xi counting the number of times event I was observed in a particular instance.



Library Used: from sklearn.naive\_bayes import MultinomialNB

**Bernoulli Naïve Bayes:**

In the multivariate [Bernoulli](https://en.wikipedia.org/wiki/Bernoulli_distribution) event model, features are independent [booleans](https://en.wikipedia.org/wiki/Boolean_data_type) (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence features are used rather than term frequencies. If xi is a Boolean expressing the occurrence or absence of the ith term from the vocabulary, then the likelihood of a document given a class {\displaystyle C\_{k}} is given by



Library Used: from sklearn.naive\_bayes import BernoulliNB

**Modelling Pipeline:**

**1.Sentence to Words:**

Tokenization is “the process of breaking a stream of text up into words, phrases, symbols, or other meaningful elements called tokens.”

Here we have tokenized each Review comment into token to model it using Naïve Bayes classifier.

Library used: tokenize and imported tokenize

**2.Lemmatize & Stemming:**

**Stemming** algorithms work by cutting off the end or the beginning of the word, taking into account a list of common prefixes and suffixes that can be found in an inflected word. This indiscriminate cutting can be successful in some occasions, but not always, and that is why we affirm that this approach presents some limitations.

Library used: Porter Stemmer from nltk.stem

**Lemmatization**, on the other hand, takes into consideration the morphological analysis of the words. To do so, it is necessary to have detailed dictionaries which the algorithm can look through to link the form back to its lemma. Again, you can see how it works with the same example words.

Library used: nltk.stem.wordnet from WordNetLemmatizer

**3.Remove Stop Words:**

**Stop Words:** A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

Library used: from wordcloud importing STOPWORDS

**4.Model Fitting:**

We fit different models to the above data

**Count Vectorizer:**

Each message is separated into tokens and number of times each token occurs in message is counted and uses this value as its weight

Library used: from sklearn.feature\_extraction.text import CountVectorizer

**Term Frequency and Inverse Document Frequency:**

Weight is statistical measure to check how important a word is to a document in a collection. With the TFIDFVectorizer the value increases proportionally to count, but is offset by the frequency of the word in the corpus. - This is the IDF (inverse document frequency part). This helps to adjust for the fact that some words appear more frequently.

Library used: from sklearn.feature\_extraction.text import TfidfTransformer

**Model Results:**

**Precision, Recall & F-1 Score:**

Logistic Regression has got the highest F-1 score among all the models for both positive and negative comments. Even the precision, Recall are high for the Logistic Models

**Accuracy:**

|  |  |  |
| --- | --- | --- |
| **No** | **Models** | **Accuracy** |
| 1 | NLTK Naïve Bayes | 61.5% |
| 2 | Multinomial | 93.2% |
| 3 | Bernouli Naïve Bayes | 92.2% |
| 4 | Logistic Regression | 93.4% |

Library Used: from sklearn.metrics import confusion\_matrix

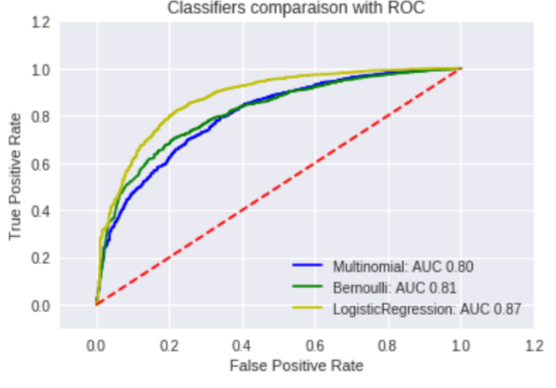
Logistic Regression has got the high accuracy out of all other models.

Multinomial, Bernoulli Naïve Bayes has got similar accuracies but bit less than the Logistic Regression. NLTK classifier has got the least accuracy 61.5%

|  |  |  |  |
| --- | --- | --- | --- |
| **Metrics** | **Multinomial NB** | **Bernouli NB** | **Logistic** |
| Precision(+ve) | 0.00 | 0.37 | 0.52 |
| Precision(-ve) | 0.93 | 0.94 | 0.95 |
| Recall(+ve) | 0.00 | 0.19 | 0.36 |
| Recall(-ve) | 1.00 | 0.98 | 0.98 |
| F1-Score(+ve) | 0.00 | 0.25 | 0.43 |
| F1-Score(-ve) | 0.96 | 0.96 | 0.97 |

**ROC Curves:**

Logistic NB Performs better than the remaining all Models. Bernoulli and Multinomial NB has the similar performance.



Library used: from sklearn.metrics import roc\_curve, auc

Area under curve for different models:

|  |  |
| --- | --- |
| **Model** | **AUC** |
| Multinomial NB | 0.80 |
| Bernoulli NB | 0.81 |
| Logistic | 0.87 |

**Bag of Words:**

Most occurring words in training set.

Coefficients for the logistic function of each feature the slow, terrible, retuning, old love are the some of the important words with negative review sentiment.



**Repeated Words:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Positive words** | **Negative Words** |
| 1 | Wanted | Amazon |
| 2 | App | Wanted |
| 3 | Tab | Good |
| 4 | Use | Use |
| 5 | Kindle | Charger |
| 6 | Love | Hoping |
| 7 | Reading | Buy |

**Frequency distribution of words-**

We have done the frequency distribution of words and analysed the trend of most repeated words.

Below are the frequent good and bad words-

Some words are repeating in both the sections showing that the context could be different in different scenario i.e. “This product is good” / “This product is not good”

**Bag of Words**

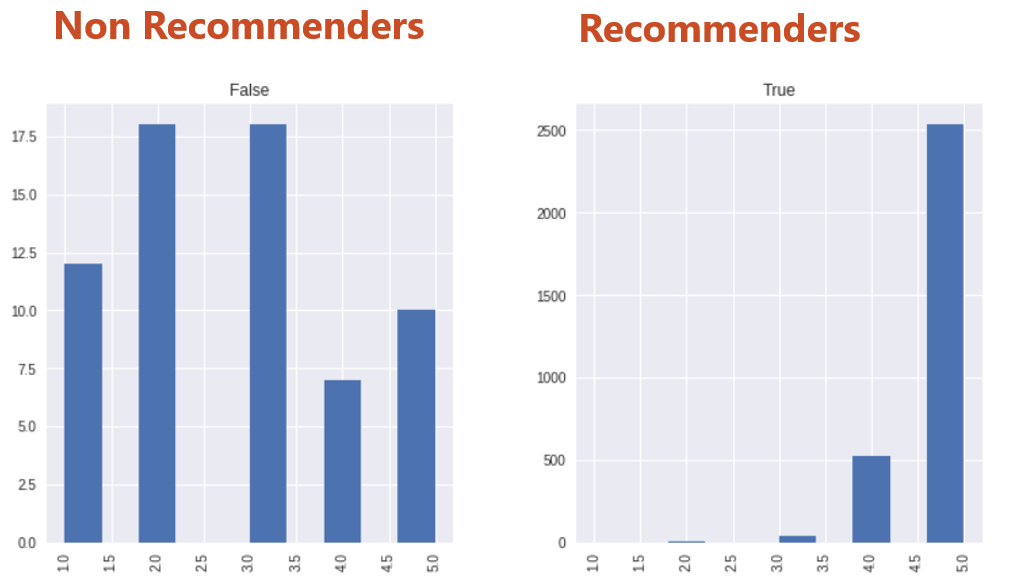


**Deep Dive of a Particular Product:**

For this analysis, we have selected “kindle” to see the distribution of ratings and its NPS score. After removing the null values, the distribution of ratings showed the following results-



Most of the users gave 5-star ratings and kindle had a NPS score of around 77%, showing most of the users were promoters. We also saw the distribution of ratings for recommenders and non-recommenders



Those who recommend amazon kindle generate high NPS score of 98.23. Those who DO NOT recommend kindle produce a NPS score of 20.0

**Business recommendations:**

This work not only helps the company to derive information from customer reviews, but presents a broader application prospects of natural language analysis as well. Based on our analysis, we provide the following recommendations.

* **On the Webpage, highlight the traits of product or keywords that have most positive effects to attract more customers.**

Those are feature that reviewers mostly satisfied with.

* **The company can market research and work on product development against the reviews that have negative effects.**

In our case, positive or negative effects can be indicated by the coefficients in the Logistic Regression Model.

* **Provide Incentives/promo code in order to encourage customers to give more reviews.**

More reviews can help us learn more about the products and improve the model.

* **Provide referral promo code in order to encourage customers to recommend the products to others.**

We can use the text analysis along with ratings and surveys to distinguish reviewers who are truly willing to recommend the products to others and increase the sales.

* **Amazon will get to know what is trending and emphasize on increasing the supply of trending product.**

Trending products usually have more reviews.

* **Amazon can analyze negative effects and improve on their customer retention model and improve customer service.**

Repeated words mean that many customers care about the same features. For instance, the word ‘Charger’ appears many times in negative reviews. It implies that the charger of the product may have a general flaw.

* **Forecast market movement based on news, blogs and social media sentiment.**

Apart from reviews on Amazon, our model can also be applied on related texts from other data sources including public social media.

**Code Links:**

Edit access is given to everyone who has the link. Even code files are shared separately.

**Different Models (open with Google Collaboratory)**

This link has the code for all different models

[**https://drive.google.com/open?id=1Gwa4mwQl8kVPIkw\_PYPwFMwBScYwHnkB**](https://drive.google.com/open?id=1Gwa4mwQl8kVPIkw_PYPwFMwBScYwHnkB)

**Deep Dive Product (open with Google Collaboratory)**

Deep dive of the product

[**https://drive.google.com/open?id=1Xx\_LleziyVhwq7mPFgZDOKrJg6tx3zc1**](https://drive.google.com/open?id=1Xx_LleziyVhwq7mPFgZDOKrJg6tx3zc1)

**Appendix:**

**Plots: Time series for reviews**

Apart from our model, we also tried making some time series plots, though we did not find it helpful to improve our model.

