AGENO SCHOOL OF BUSINESS

Golden Gate University

MSBA 327- Text Analytics

Xiaowei Guan, Ph.D.

Fall 2020

Text Analytics on Customer's Reviews on Amazon Products

Fig:1

Buse Bastug, SID: 0598594
Reena Sehitya, SID: 0597218
12/20/2020

Abstract

The aim of our project is to explore and thoroughly analyze text reviews and ratings posted by customers about the price and various products that are being sold by Amazon on its website. In order to run our text analysis successfully, we are using powerful natural language processing algorithms available in IBM SPSS Modeler and Python Jupyter Notebook. We agreed to work on this project as we were passionate to learn and analyze the text reviews posted by customers to understand how they feel, talk and write about the products they are buying from Amazon. Such analysis will help us in finding out the influential effects of the reviews on the purchasing decisions made by others after reading reviews posted for each product. In order to accomplish our goals, we are going to run text summarization, classification, clustering, building concepts and sentiment analysis. In terms of sentiment analysis, we are going to train a model based on sentiment labels and text reviews given by customers for Amazon products using classification algorithm so that model can identify sentiments on its own when new instances of reviews are posted on amazon with higher precision.

Table of Content

Introduction

Internet has changed almost every human interaction and various aspects of daily life. Shopping is also one of the aspects in human's life that has been around since the beginning of time. Thanks to the massive growth in number of products that people can immediately access and the convenience of buying things online; globally online shopping is now delivering the dual benefits of comfort and easy os executing a trade online. Another major change occurred due to the emergence of the internet is the research and analysis part that is called "customer reviews". Information for any product became very handy that now customers can make better decisions by checking all the reviews from real people and get a better understanding of each products without being under the influence of any salesperson.

In this regard, Amazon is one of the first platform that was started to sell products online and one of the largest companies in the world. Amazon has consistently been growing as a company with more products that can be purchased online and product reviews being available for those products. Consumers are posting reviews directly on product pages in real time. With the vast amount of consumer reviews, this creates an opportunity to see how the market reacts to a specific product (Zhang, 2020).

The following paper will present text summary and categorization combined with sentiment analysis in order to understand customer satisfaction with Amazon products. The overall objective of this study is to understand the possibility of positive and negative sentiments for each product in the dataset and to predict positive, neutral, or negative sentiments of the reviews posted by customers for the products that they purchased on Amazon. Such prediction and deep analysis of text reviews and ratings will help the Amazon to understand as to what people think and feel about the product, brands, services and the online buying experience on Amazon.

Datasets

The datasets that we are using in our paper is "Consumer Reviews of Amazon Products". It is a publicly available datasets on Kaggle.com and originally provided Datafinit's Product Database. Some of the fields of our datasets had null values that we dealt by removing them in the data cleaning and pre-processing section. We are using three datasets for our project that provide basic product information such as rating, review text, brand, categories and many more for each product. Three of the datasets are in csv format.

Name of our datasets are:

- 1429_1.csv
- Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv
- Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_May19

There are in total 38,000 rows and 21 columns but we are focusing on three columns for our analysis and they are:

- Id: Id represents the products that are sold by Amazon
- reviews.text: This column represents reviews given by customers for each product they bought online
- reviews.rating: This column represents ratings given by customers for each product they bought

In the entire paper, our focus will be on analyzing the reviews.text as well as reviews.rating columns/fields to extract business insights and help in building predictive model based classification machine learning algorithms that can predict sentiments with as highest precision and accuracy as possible.

Analysis Flow

This section will begin first by explaining about the mentioning methodology that we have used in this paper. Later in this section of "Analysis Flow" we will explain each methodology in detail followed by result, conclusion and recommendation sections.

Methodology (Reena)

In order to analyze our datasets, we have run text analytics methodologies as mentioned in the table below. The detail process of implementing these methodologies are explained in the next section.

Methodology	**Purpose**	**Tool**

• *Text Summarization and Modeling*	To get concepts and domain types of these concepts	IBM SPSS Modeler
• *Text Classification and Building Concepts*	To arrive different categories to understands different reaction of different customers for the products	IBM SPSS Modeler
• *Clustering and Text Link Analysis*	To build clusters to determine the identifiers in the data that would lead to a positive or negative review	IBM SPSS Modeler
• *Sentiment Analysis*	To build a predictive model that can predict positive, negative and neutral sentiments with higher accuracy and precision on new instances of text reviews posted by customers for each product.	Python Jupyter Notebook

Table 1

Text Summarization and Modeling (Buse)

The dataset was loaded into IBM SPSS Modeler by selecting Excel icon and importing as XLXS file. And then the stream was built. The analysis used a File List Node to read text from the reviews of our dataset. After connecting the File List node to Table node, a section of the results appeared as Figure 1.

A Text Model algorithm was used by using Text Mining node and a new model has been generated which named reviews.text and it has been placed on the stream canvas. (Figure 2)

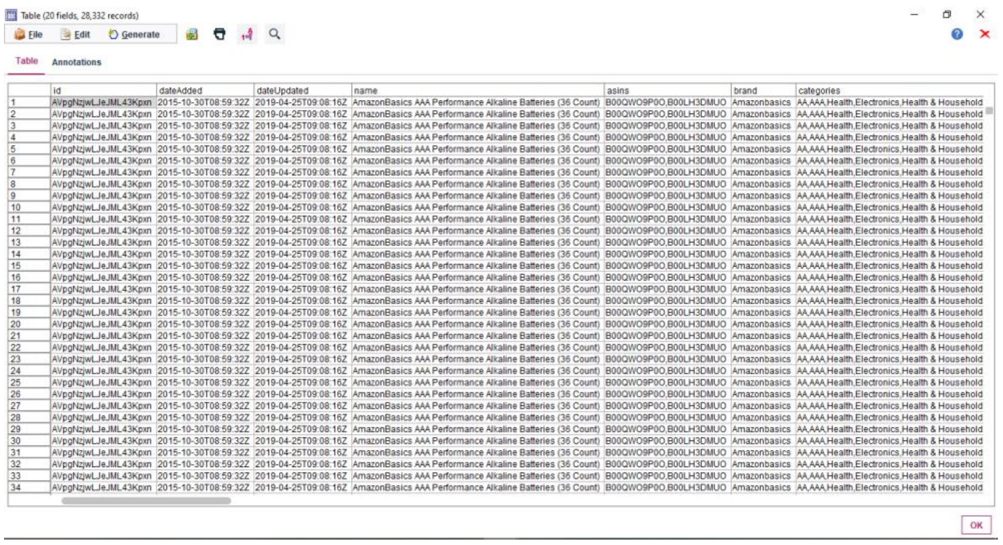
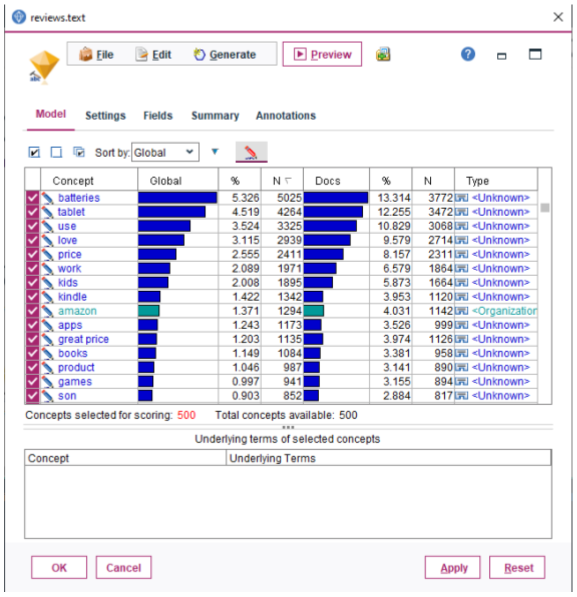


Table (20 fields, 28,332 records)

	id	dateAdded	dateUpdated	name	asins	brand	categories
1	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
2	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
3	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
4	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
5	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
6	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
7	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
8	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
9	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
10	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
11	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
12	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
13	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
14	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
15	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
16	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
17	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
18	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
19	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
20	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
21	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
22	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
23	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
24	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
25	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
26	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
27	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
28	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
29	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
30	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
31	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
32	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
33	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household
34	AVpgNzjwLJeJML43Kpxn	2015-10-30T08:59:32Z	2019-04-25T09:08:16Z	AmazonBasics AAA Performance Alkaline Batteries (36 Count)	B00QWO9P0O,B00LH3DMUO	Amazonbasics	AA,AAA,Health,Electronics,Health & Household

Figure 2: (Total of 28,332 records has been used in the analysis)


reviews.text

Sort by: Global

Concept	Global	%	N	Docs	%	N	Type
batteries		5.326	5025		13.314	3772	<Unknown>
tablet		4.519	4264		12.255	3472	<Unknown>
use		3.524	3325		10.829	3068	<Unknown>
love		3.115	2939		9.579	2714	<Unknown>
price		2.555	2411		8.157	2311	<Unknown>
work		2.089	1971		6.579	1864	<Unknown>
kids		2.008	1895		5.873	1664	<Unknown>
kindle		1.422	1342		3.953	1120	<Unknown>
amazon		1.371	1294		4.031	1142	<Organizatior
apps		1.243	1173		3.526	999	<Unknown>
great price		1.203	1135		3.974	1126	<Unknown>
books		1.149	1084		3.381	958	<Unknown>
product		1.046	987		3.141	890	<Unknown>
games		0.997	941		3.155	894	<Unknown>
son		0.903	852		2.884	817	<Unknown>

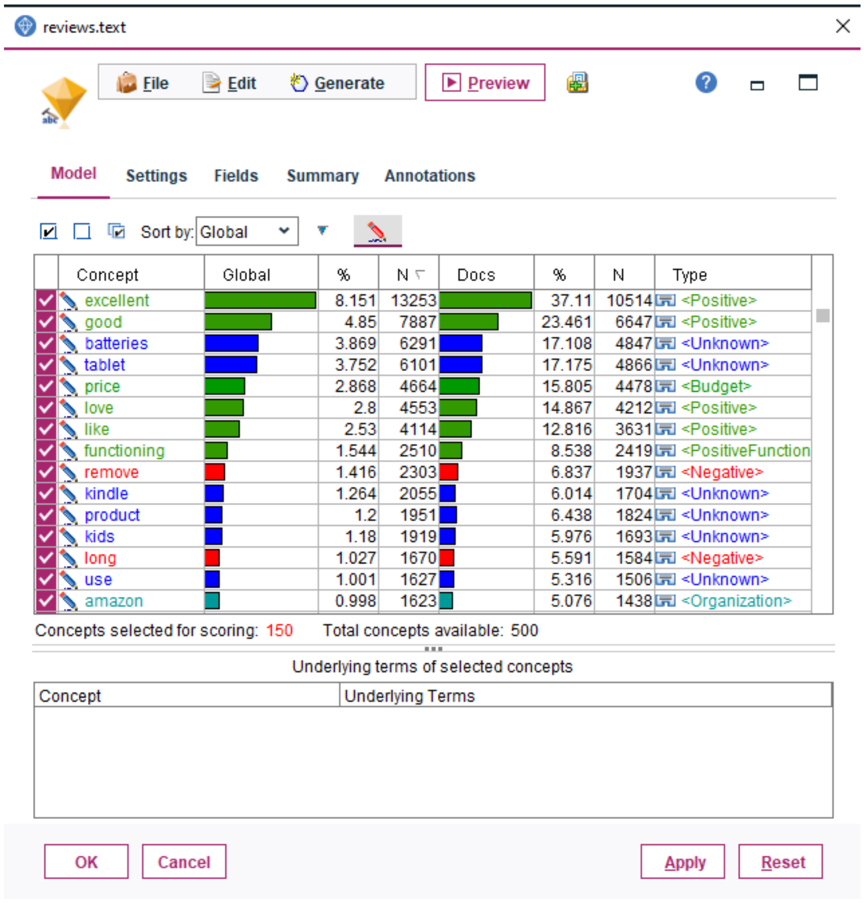
Concepts selected for scoring: 500 Total concepts available: 500

Underlying terms of selected concepts

Concept	Underlying Terms

Figure 3: Model Output

Generated model shows 500 selected concepts and domain types of these concepts labelled as <Unknown>. Since the first generated model does not provide a very good insight of the dataset, another text mining concept has been created by checking concepts based on highest frequency and Opinions (English) text analysis package. Our new generated model contains 150 total concepts that has been scored and ordered by global frequency. Also listed is the document/record frequency. As is typical when using a resources template that has not been tuned to this particular text dataset, there are some concepts with Unknown type. Also, the two most common concepts are “excellent” and “good which means the reviews are mostly in positive skew with positive type of price concept. They both appear in 23-37% of all records. (Figure 4)



Sort by: Global

	Concept	Global	%	N	Docs	%	N	Type
✓	excellent	🟩	8.151	13253	🟩	37.11	10514	<Positive>
✓	good	🟩	4.85	7887	🟩	23.461	6647	<Positive>
✓	batteries	🟦	3.869	6291	🟦	17.108	4847	<Unknown>
✓	tablet	🟦	3.752	6101	🟦	17.175	4866	<Unknown>
✓	price	🟩	2.868	4664	🟩	15.805	4478	<Budget>
✓	love	🟩	2.8	4553	🟩	14.867	4212	<Positive>
✓	like	🟩	2.53	4114	🟩	12.816	3631	<Positive>
✓	functioning	🟩	1.544	2510	🟩	8.538	2419	<PositiveFunction
✓	remove	🟥	1.416	2303	🟥	6.837	1937	<Negative>
✓	kindle	🟦	1.264	2055	🟦	6.014	1704	<Unknown>
✓	product	🟦	1.2	1951	🟦	6.438	1824	<Unknown>
✓	kids	🟦	1.18	1919	🟦	5.976	1693	<Unknown>
✓	long	🟥	1.027	1670	🟥	5.591	1584	<Negative>
✓	use	🟦	1.001	1627	🟦	5.316	1506	<Unknown>
✓	amazon	🟦	0.998	1623	🟦	5.076	1438	<Organization>

Concepts selected for scoring: 150 Total concepts available: 500

Underlying terms of selected concepts

Concept	Underlying Terms

Figure 4: Generated Model Output

Text Classification and Building Concepts (Buse)

Due to size of the dataset, our classification model has been built by 25% Random Sampling technique. (Figure 5)

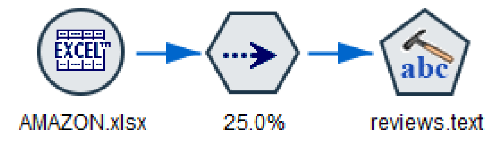
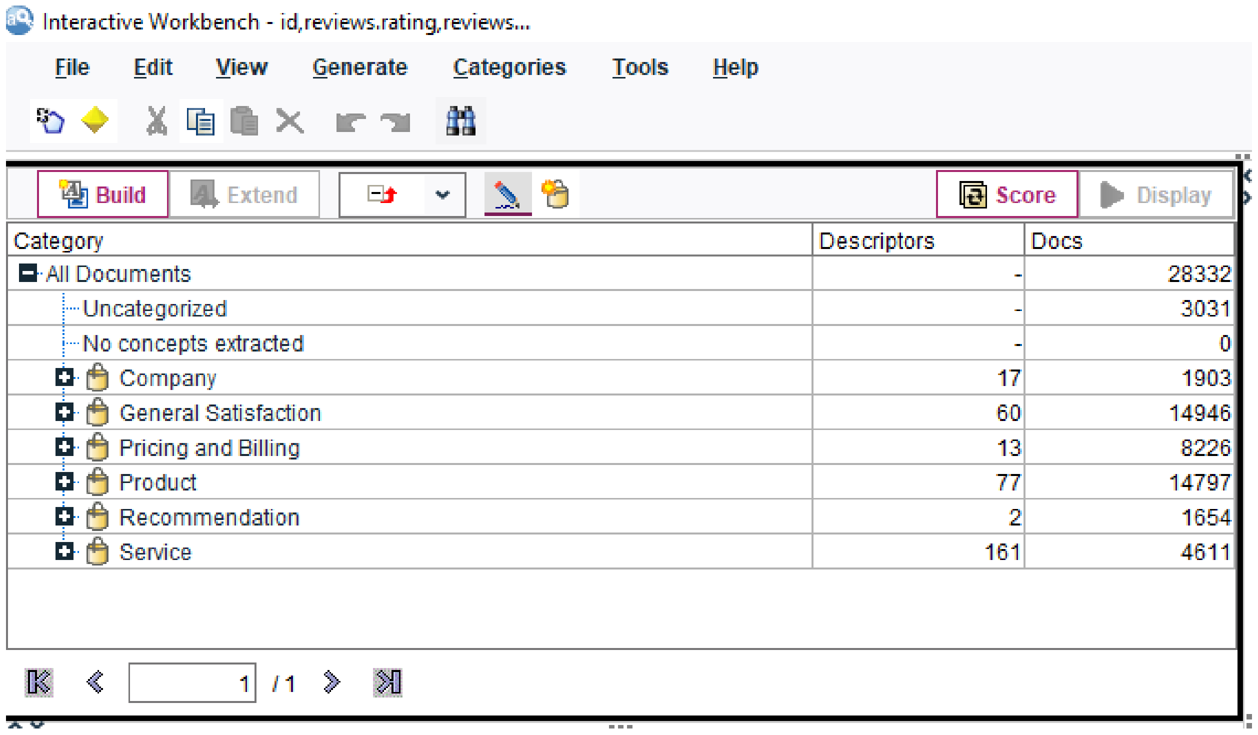


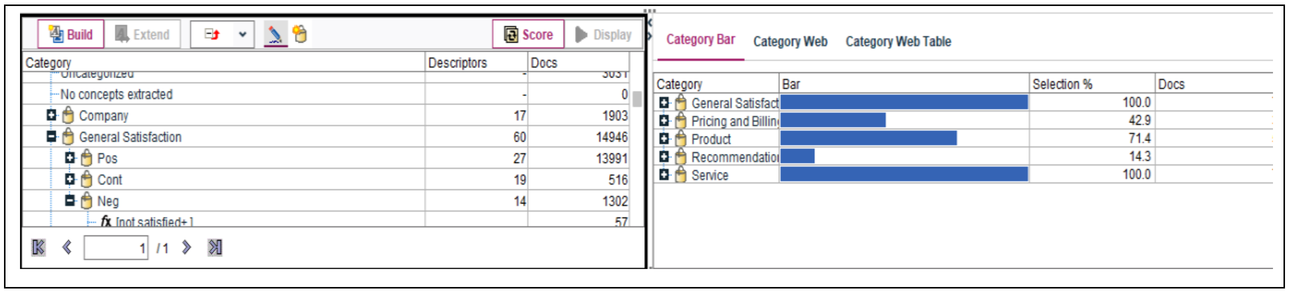
Figure 5: Classification model

After grouping and scoring all categories by using Interactive Workbench panel, the result shows that uncategorized documents are 3031 and the dataset has 6 different categories. In order to understand the customers' reactions about what they most like & dislike of the products, we can focus on general satisfaction and pricing & billing concepts. (Figure 5)


Interactive Workbench - id,reviews.rating,reviews...
File Edit View Generate Categories Tools Help
Build Extend Score Display
1 / 1

Category	Descriptors	Docs
All Documents	-	28332
Uncategorized	-	3031
No concepts extracted	-	0
Company	17	1903
General Satisfaction	60	14946
Pricing and Billing	13	8226
Product	77	14797
Recommendation	2	1654
Service	161	4611

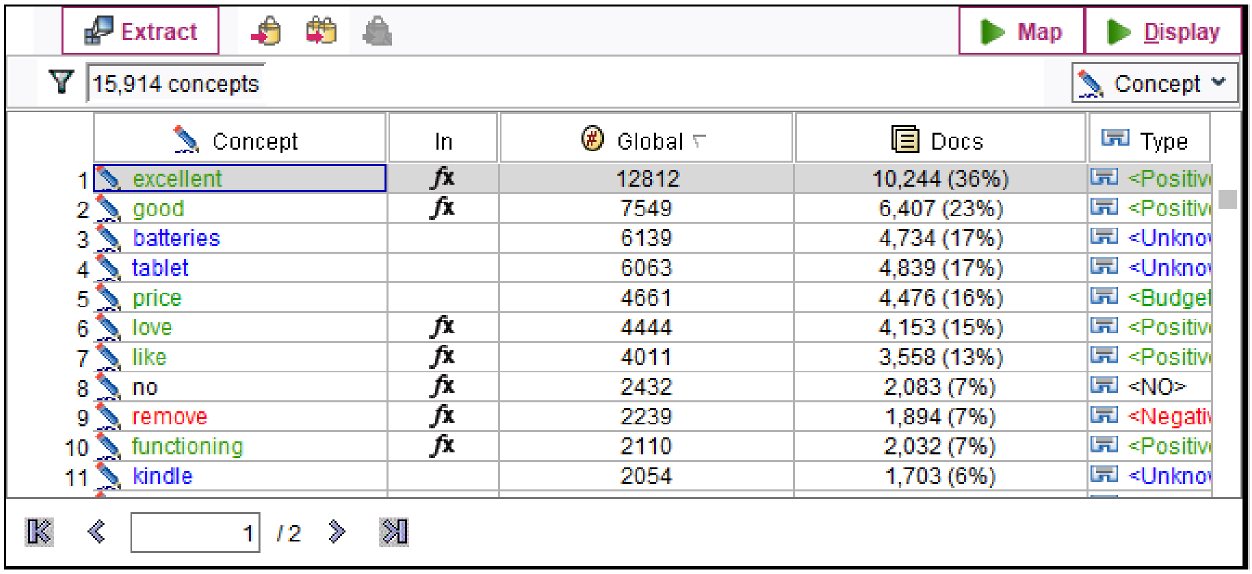
Figure 6: Grouping



Category	Descriptors	Docs
Uncategorized	-	3031
No concepts extracted	-	0
Company	17	1903
General Satisfaction	60	14946
Pos	27	13991
Cont	19	516
Neg	14	1302
fx [not satisfied+]		57

Category	Bar	Selection %	Docs
General Satisfact		100.0	
Pricing and Billin		42.9	
Product		71.4	
Recommendatio		14.3	
Service		100.0	

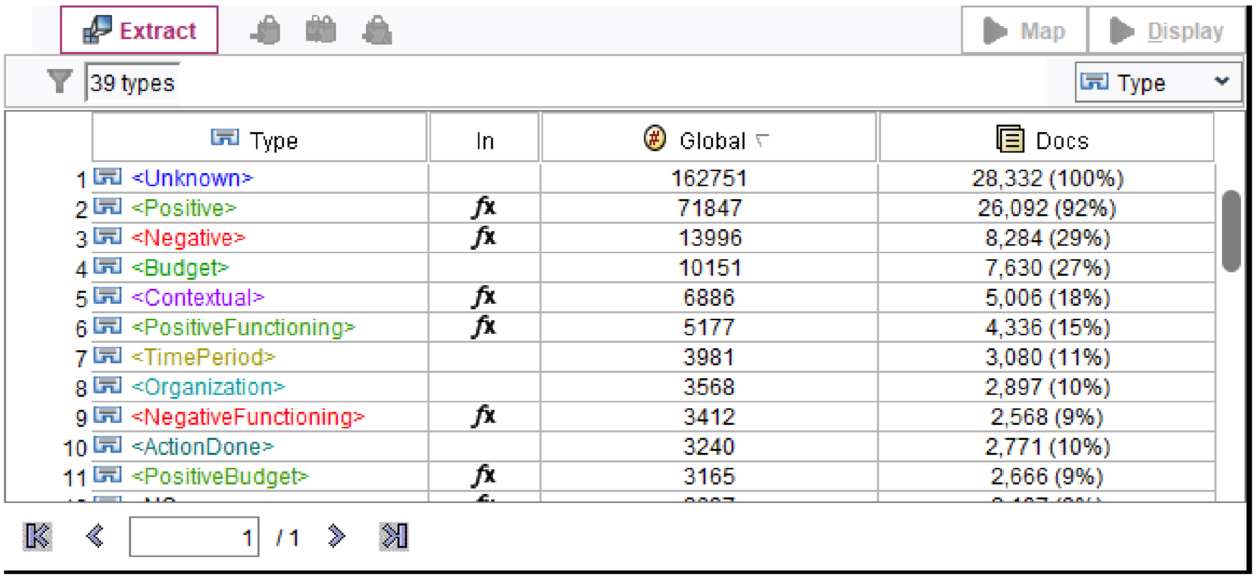
Figure 7: Categories by selected %



	Concept	In	Global	Docs	Type
1	excellent	*fx*	12812	10,244 (36%)	<Positiv
2	good	*fx*	7549	6,407 (23%)	<Positiv
3	batteries		6139	4,734 (17%)	<Unkno
4	tablet		6063	4,839 (17%)	<Unkno
5	price		4661	4,476 (16%)	<Budget
6	love	*fx*	4444	4,153 (15%)	<Positiv
7	like	*fx*	4011	3,558 (13%)	<Positiv
8	no	*fx*	2432	2,083 (7%)	<NO>
9	remove	*fx*	2239	1,894 (7%)	<Negati
10	functioning	*fx*	2110	2,032 (7%)	<Positiv
11	kindle		2054	1,703 (6%)	<Unkno

Figure 8

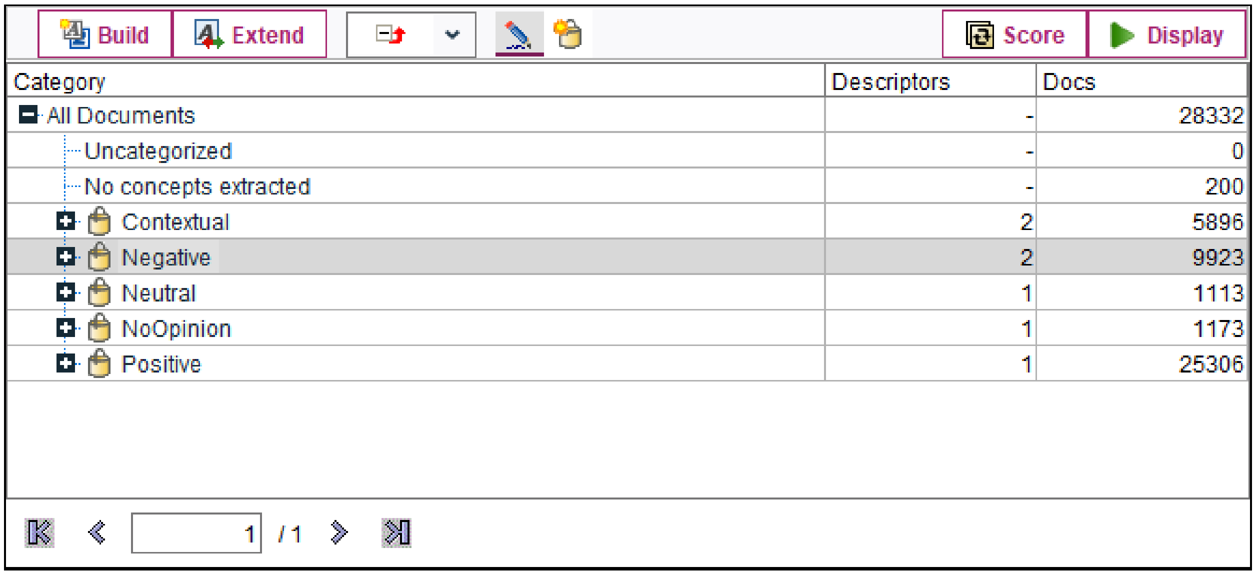
The most popular concepts are: excellent, tablet, good and battery. Excellent and good show up in 36% and 23% of documents. The most popular types of documents are positive and negative functioning. It shows us positive docs are 92% while the negative docs are only 29%(Figure 9)



	Type	In	Global	Docs
1	<Unknown>		162751	28,332 (100%)
2	<Positive>	*fx*	71847	26,092 (92%)
3	<Negative>	*fx*	13996	8,284 (29%)
4	<Budget>		10151	7,630 (27%)
5	<Contextual>	*fx*	6886	5,006 (18%)
6	<PositiveFunctioning>	*fx*	5177	4,336 (15%)
7	<TimePeriod>		3981	3,080 (11%)
8	<Organization>		3568	2,897 (10%)
9	<NegativeFunctioning>	*fx*	3412	2,568 (9%)
10	<ActionDone>		3240	2,771 (10%)
11	<PositiveBudget>	*fx*	3165	2,666 (9%)

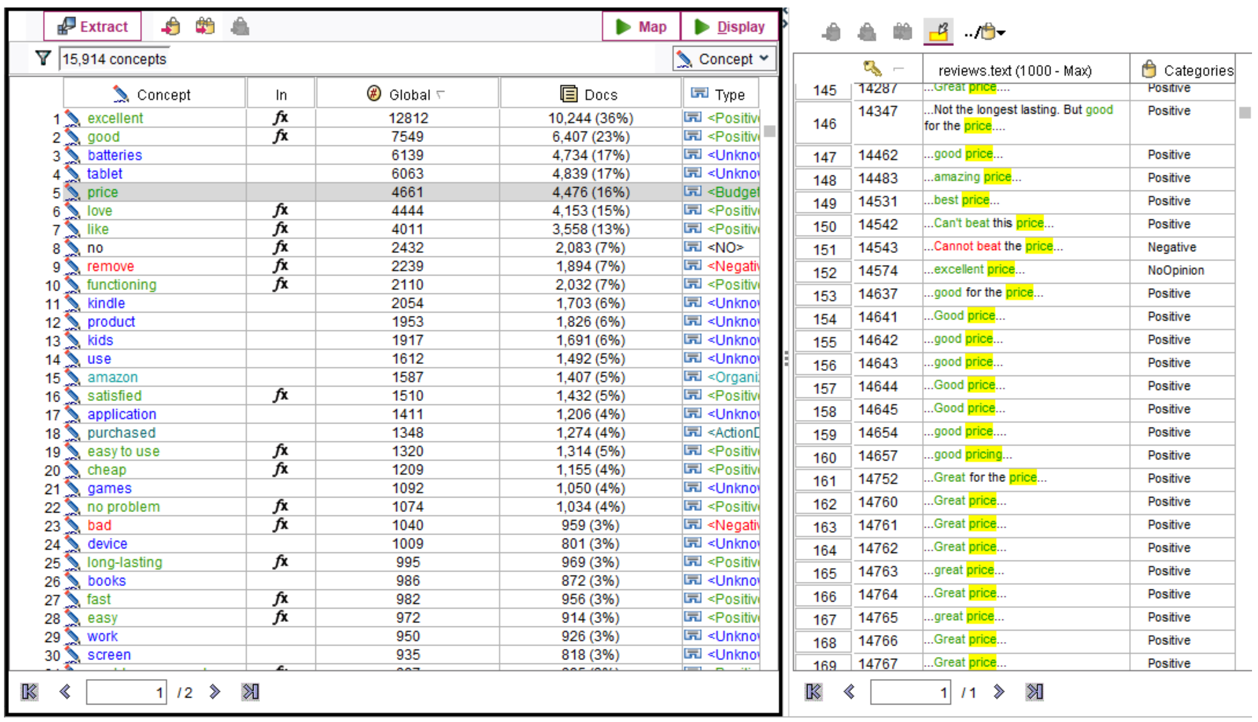
Figure 9

After scoring the records as contextual, negative, neutral, no opinion and positive we get the scores as seen below image. (Figure 10)



Category	Descriptors	Docs
All Documents	-	28332
Uncategorized	-	0
No concepts extracted	-	200
Contextual	2	5896
Negative	2	9923
Neutral	1	1113
NoOpinion	1	1173
Positive	1	25306

Figure 10



15,914 concepts

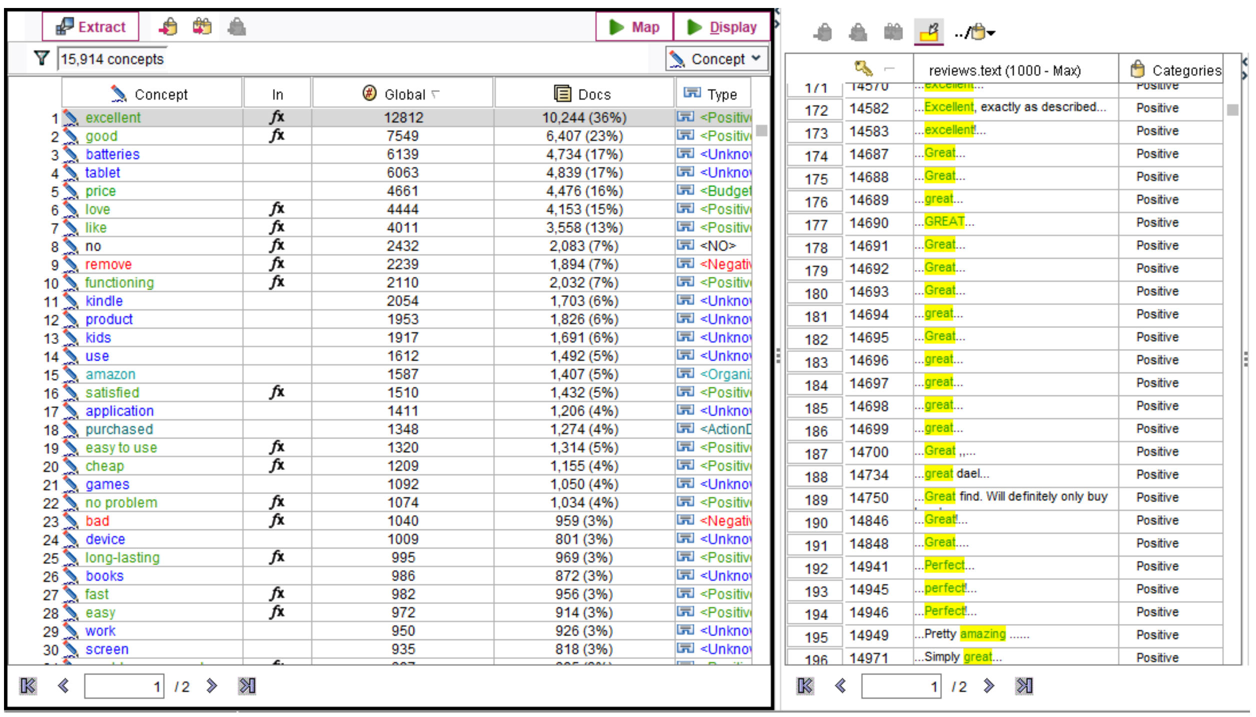
	Concept	In	Global	Docs	Type
1	excellent	*fx*	12812	10,244 (36%)	<Positiv
2	good	*fx*	7549	6,407 (23%)	<Positiv
3	batteries		6139	4,734 (17%)	<Unknov
4	tablet		6063	4,839 (17%)	<Unknov
5	price		4661	4,476 (16%)	<Budget
6	love	*fx*	4444	4,153 (15%)	<Positiv
7	like	*fx*	4011	3,558 (13%)	<Positiv
8	no	*fx*	2432	2,083 (7%)	<NO>
9	remove	*fx*	2239	1,894 (7%)	<Negati
10	functioning	*fx*	2110	2,032 (7%)	<Positiv
11	kindle		2054	1,703 (6%)	<Unknov
12	product		1953	1,826 (6%)	<Unknov
13	kids		1917	1,691 (6%)	<Unknov
14	use		1612	1,492 (5%)	<Unknov
15	amazon		1587	1,407 (5%)	<Organi
16	satisfied	*fx*	1510	1,432 (5%)	<Positiv
17	application		1411	1,206 (4%)	<Unknov
18	purchased		1348	1,274 (4%)	<ActionD
19	easy to use	*fx*	1320	1,314 (5%)	<Positiv
20	cheap	*fx*	1209	1,155 (4%)	<Positiv
21	games		1092	1,050 (4%)	<Unknov
22	no problem	*fx*	1074	1,034 (4%)	<Positiv
23	bad	*fx*	1040	959 (3%)	<Negati
24	device		1009	801 (3%)	<Unknov
25	long-lasting	*fx*	995	969 (3%)	<Positiv
26	books		986	872 (3%)	<Unknov
27	fast	*fx*	982	956 (3%)	<Positiv
28	easy	*fx*	972	914 (3%)	<Positiv
29	work		950	926 (3%)	<Unknov
30	screen		935	818 (3%)	<Unknov

1 / 2

		reviews.text (1000 - Max)	Categories
145	14287	...Great price....	Positive
146	14347	...Not the longest lasting. But good for the price....	Positive
147	14462	...good price...	Positive
148	14483	...amazing price...	Positive
149	14531	...best price...	Positive
150	14542	...Can't beat this price...	Positive
151	14543	...Cannot beat the price...	Negative
152	14574	...excellent price...	NoOpinion
153	14637	...good for the price...	Positive
154	14641	...Good price...	Positive
155	14642	...good price...	Positive
156	14643	...good price...	Positive
157	14644	...Good price...	Positive
158	14645	...Good price...	Positive
159	14654	...good price....	Positive
160	14657	...good pricing...	Positive
161	14752	...Great for the price...	Positive
162	14760	...Great price...	Positive
163	14761	...Great price...	Positive
164	14762	...Great price...	Positive
165	14763	...great price...	Positive
166	14764	...Great price...	Positive
167	14765	...great price...	Positive
168	14766	...Great price...	Positive
169	14767	...Great price...	Positive

1 / 1

It is still shows that our dataset has more positive records than other concepts. The main point in this analysis will be focusing on negative reviews and to detect what key words of it. Customers are generally very happy with the product they purchase.



15,914 concepts

	Concept	In	Global	Docs	Type
1	excellent	*fx*	12812	10,244 (36%)	<Positiv
2	good	*fx*	7549	6,407 (23%)	<Positiv
3	batteries		6139	4,734 (17%)	<Unknov
4	tablet		6063	4,839 (17%)	<Unknov
5	price		4661	4,476 (16%)	<Budget
6	love	*fx*	4444	4,153 (15%)	<Positiv
7	like	*fx*	4011	3,558 (13%)	<Positiv
8	no	*fx*	2432	2,083 (7%)	<NO>
9	remove	*fx*	2239	1,894 (7%)	<Negati
10	functioning	*fx*	2110	2,032 (7%)	<Positiv
11	kindle		2054	1,703 (6%)	<Unknov
12	product		1953	1,826 (6%)	<Unknov
13	kids		1917	1,691 (6%)	<Unknov
14	use		1612	1,492 (5%)	<Unknov
15	amazon		1587	1,407 (5%)	<Organi
16	satisfied	*fx*	1510	1,432 (5%)	<Positiv
17	application		1411	1,206 (4%)	<Unknov
18	purchased		1348	1,274 (4%)	<ActionD
19	easy to use	*fx*	1320	1,314 (5%)	<Positiv
20	cheap	*fx*	1209	1,155 (4%)	<Positiv
21	games		1092	1,050 (4%)	<Unknov
22	no problem	*fx*	1074	1,034 (4%)	<Positiv
23	bad	*fx*	1040	959 (3%)	<Negati
24	device		1009	801 (3%)	<Unknov
25	long-lasting	*fx*	995	969 (3%)	<Positiv
26	books		986	872 (3%)	<Unknov
27	fast	*fx*	982	956 (3%)	<Positiv
28	easy	*fx*	972	914 (3%)	<Positiv
29	work		950	926 (3%)	<Unknov
30	screen		935	818 (3%)	<Unknov

1 / 2

		reviews.text (1000 - Max)	Categories
171	14570	...excellent...	Positive
172	14582	...Excellent, exactly as described...	Positive
173	14583	...excellent!...	Positive
174	14687	...Great...	Positive
175	14688	...Great...	Positive
176	14689	...great...	Positive
177	14690	...GREAT...	Positive
178	14691	...Great...	Positive
179	14692	...Great...	Positive
180	14693	...Great...	Positive
181	14694	...great...	Positive
182	14695	...Great...	Positive
183	14696	...great...	Positive
184	14697	...great...	Positive
185	14698	...great...	Positive
186	14699	...great...	Positive
187	14700	...Great ,,...	Positive
188	14734	...great dael...	Positive
189	14750	...Great find. Will definitely only buy	Positive
190	14846	...Great!...	Positive
191	14848	...Great....	Positive
192	14941	...Perfect...	Positive
193	14945	...perfect!...	Positive
194	14946	...Perfect!...	Positive
195	14949	...Pretty amazing	Positive
196	14971	...Simply great...	Positive

1 / 2

Checking the price section, we see the relation with positive type and the texts. Mostly, price of the products meet the customers' expectations.

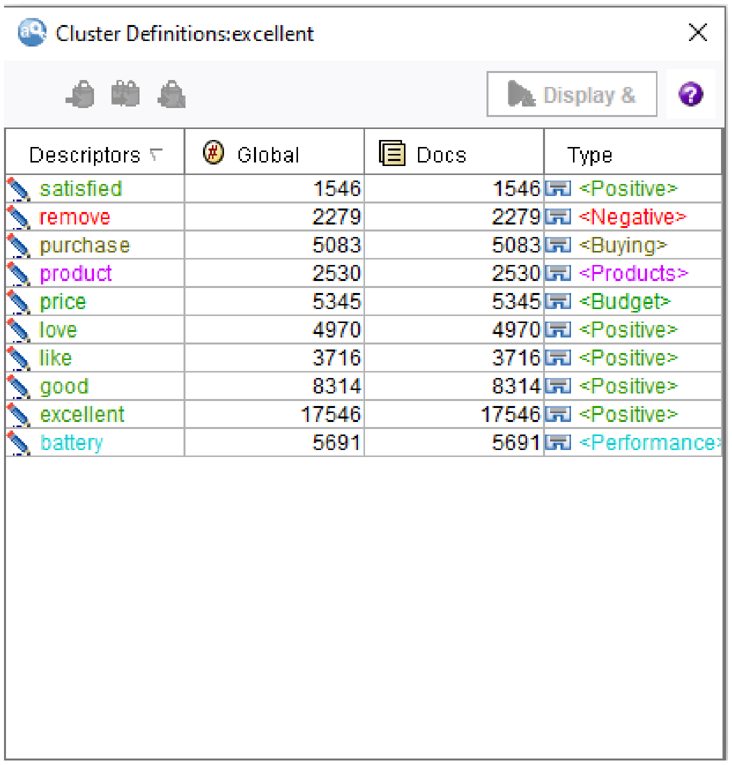
Clustering and Text Link Analysis (Buse)

Clusters were built in order to determine the identifiers in the data that would lead to a positive or negative review.

Clusters were built utilizing these parameters:

- Top number of concepts
- Number based on document count: 25,000
- Maximum number of clusters to create: 500
- Maximum concepts in a cluster: 20
- Minimum concepts in a cluster: 5
- Maximum number of internal links: 20
- Maximum number of external links: 20
- Minimum link value: 5

After only choosing Budget, Buying, Positive, Negative, Performance and Product Types, ten clusters were defined as it can be checked below. (Figure 11)



Cluster Definitions:excellent

Descriptors	Global	Docs	Type
satisfied	1546	1546	<Positive>
remove	2279	2279	<Negative>
purchase	5083	5083	<Buying>
product	2530	2530	<Products>
price	5345	5345	<Budget>
love	4970	4970	<Positive>
like	3716	3716	<Positive>
good	8314	8314	<Positive>
excellent	17546	17546	<Positive>
battery	5691	5691	<Performance

Figure 11

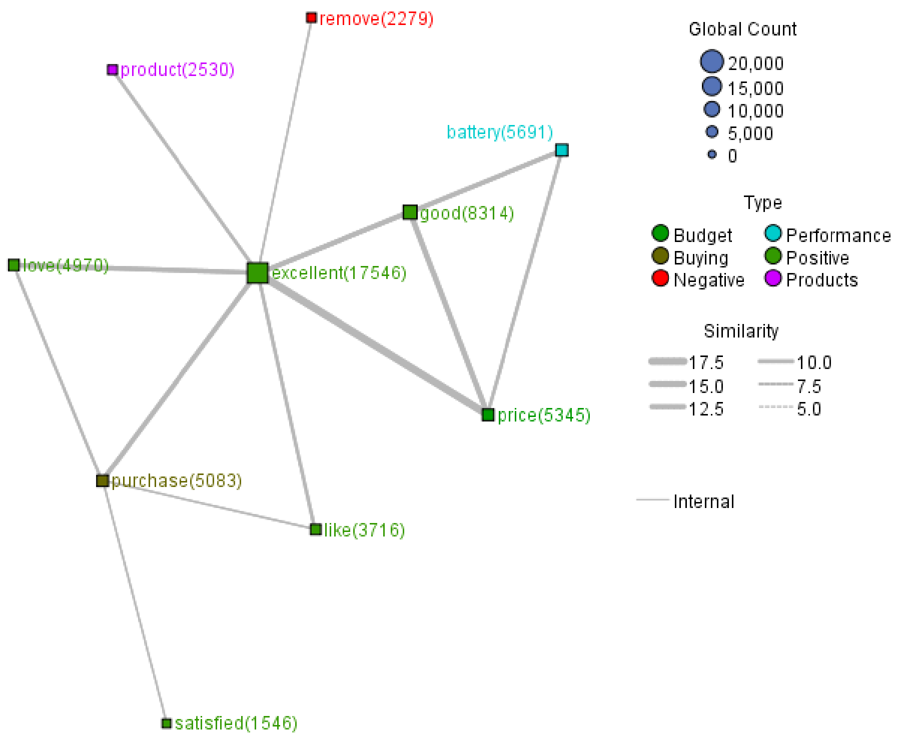
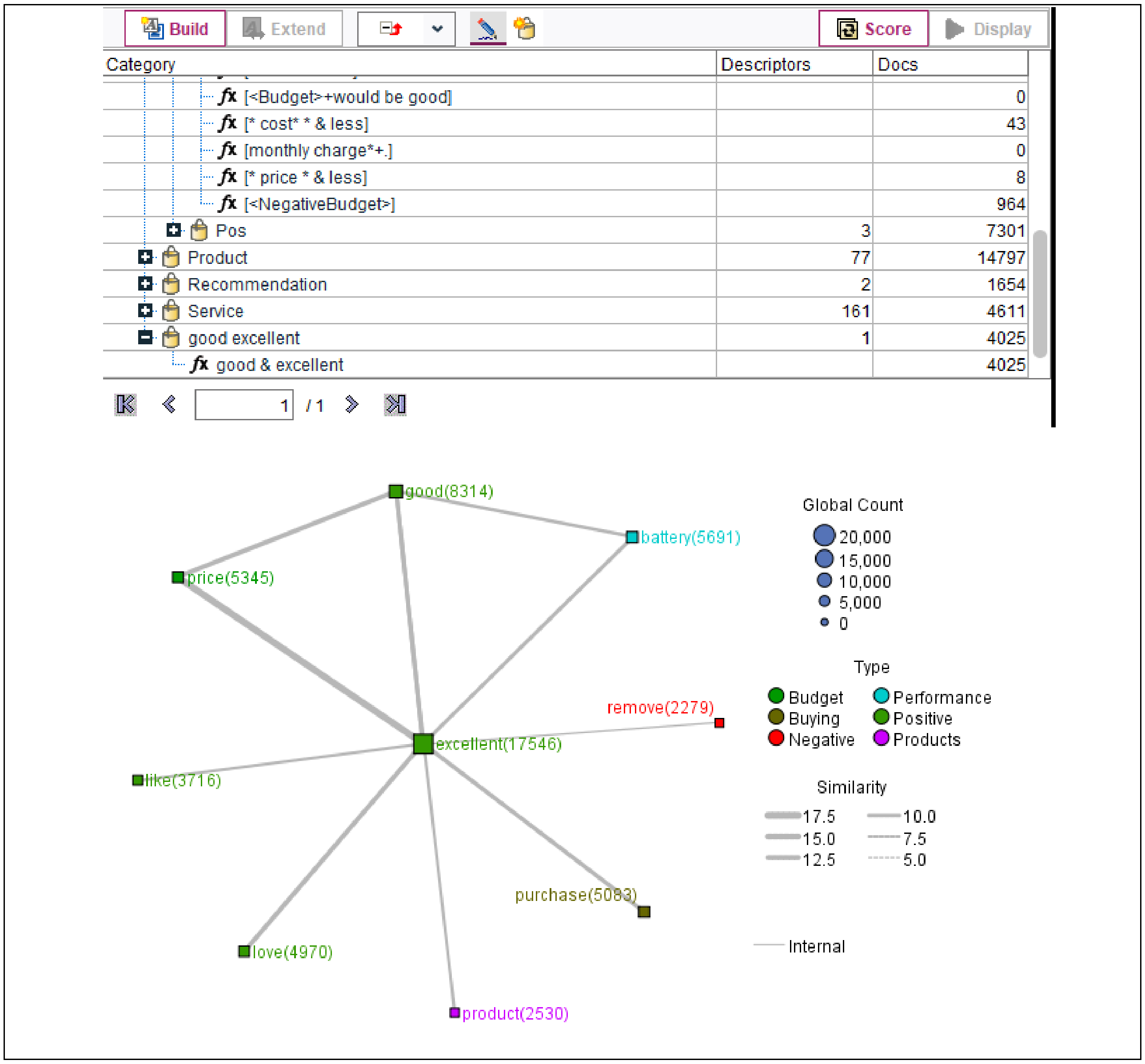


Figure 12: Concept Web

As it can be seen above, excellent is connected with price, purchase and product. For all these clusters, no negative relationships were actually found. Overall, IBM SPSS Modeler showed that there were more positive reviews than negative. When customers were mentioning “excellent” in their reviews for 18,000 times, “good” was mentioned for 8,000 times respectively. It means that certain customer responses contained the word "excellent" or "good" but not both. In order to check how many times both concepts occurred together, a new category was issued under “good excellent”. After scoring the categories and concepts tab one more time, we are able to check that both concepts has been occurred in 4025 docs. (Figure 13)



Category	Descriptors	Docs
[<Budget>+would be good]		0
[* cost* * & less]		43
[monthly charge*+.]		0
[* price * & less]		8
[<NegativeBudget>]		964
Pos	3	7301
Product	77	14797
Recommendation	2	1654
Service	161	4611
good excellent	1	4025
good & excellent		4025

Figure 13: Good & Excellent Concepts

We performed text analysis by adding Text Mining Node to the stream and connecting to our file. This time our file have only Amazon ID, review rates and review text. After starting the session with selecting customer satisfaction tap from the Text Analysis package, our model has been generated.(Figure 9) The analysis was focused products and budget concepts with their negative and positive relationships that could be valuable and effective for customers' reviews.

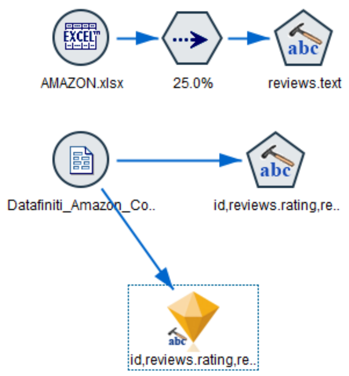
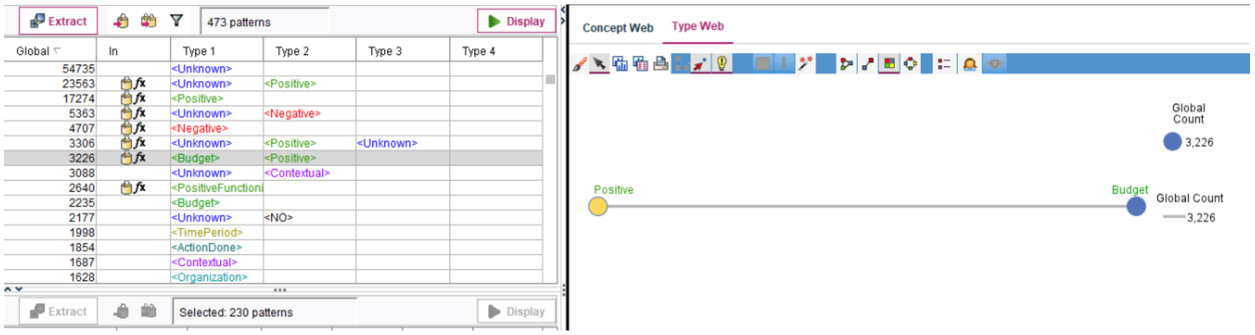


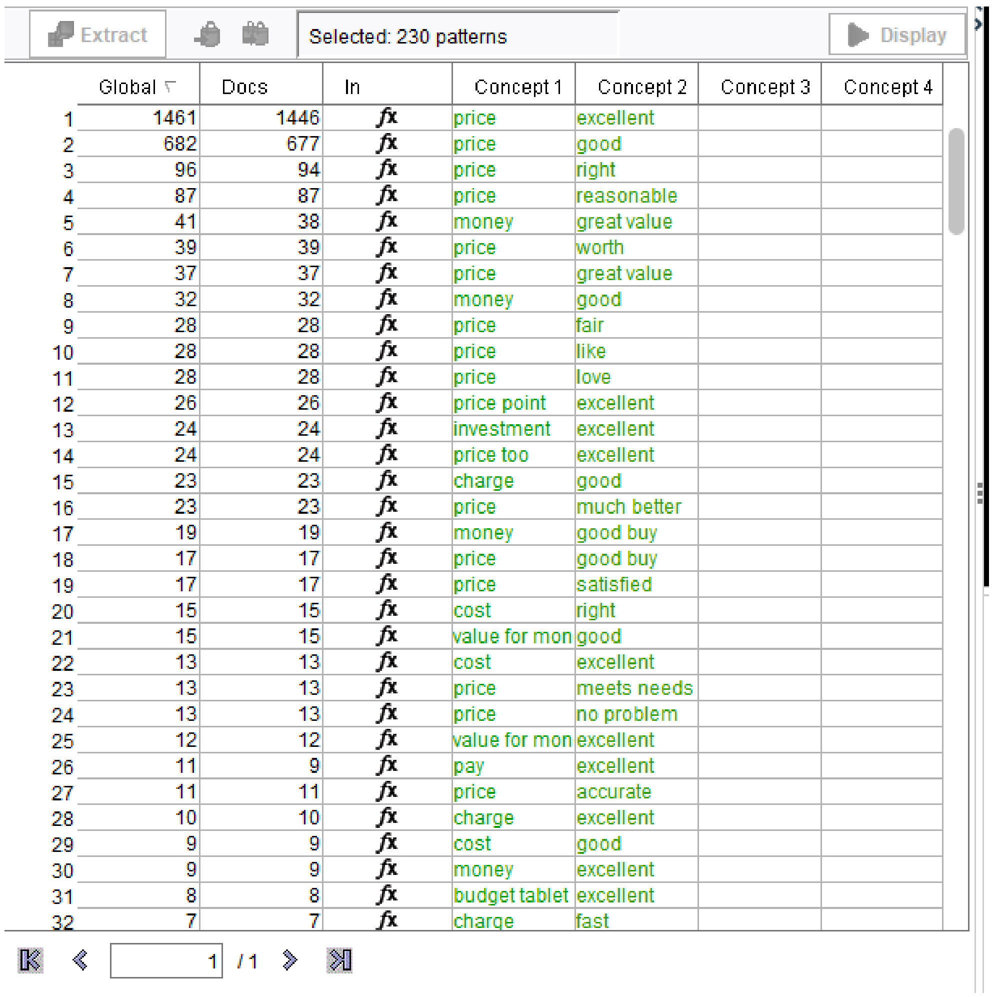
Figure 14: The model output

Looking at the Text Link Analysis, we wanted to see how budget have a relationship with positive and negative reviews.



Selected: 230 patterns

Global	In	Type 1	Type 2	Type 3	Type 4
54735		<Unknown>			
23563	fx	<Unknown>	<Positive>		
17274	fx	<Positive>			
5363	fx	<Unknown>	<Negative>		
4707	fx	<Negative>			
3306	fx	<Unknown>	<Positive>	<Unknown>	
3226	fx	<Budget>	<Positive>		
3088		<Unknown>	<Contextual>		
2640	fx	<PositiveFunctioni			
2235		<Budget>			
2177		<Unknown>	<NO>		
1998		<TimePeriod>			
1854		<ActionDone>			
1687		<Contextual>			
1628		<Organization>			

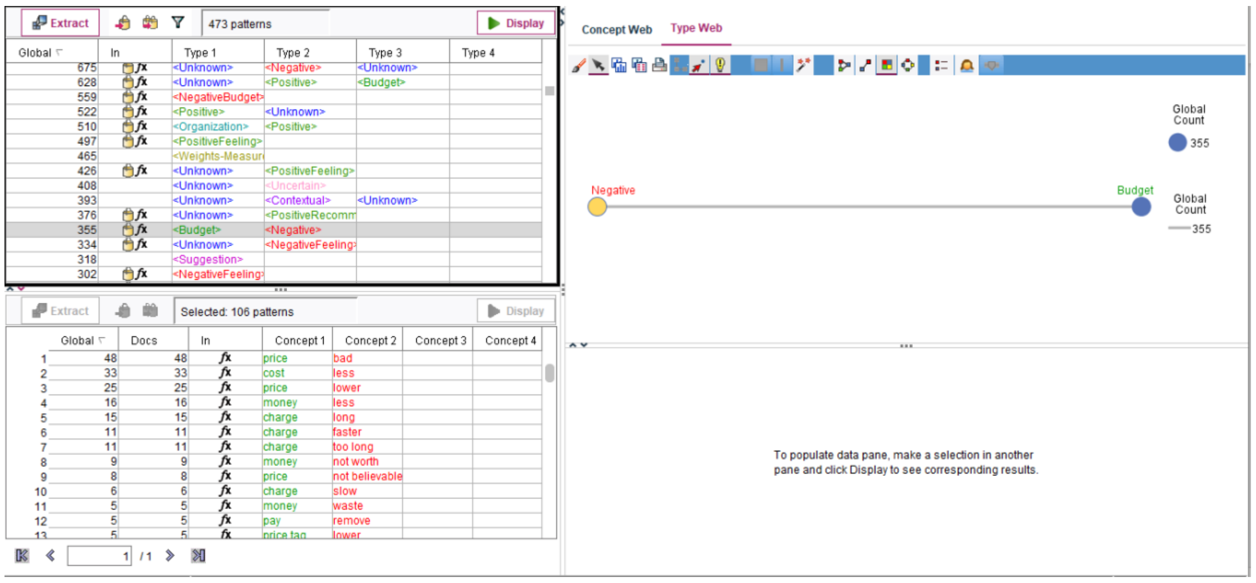


Selected: 230 patterns

	Global	Docs	In	Concept 1	Concept 2	Concept 3	Concept 4
1	1461	1446	*fx*	price	excellent		
2	682	677	*fx*	price	good		
3	96	94	*fx*	price	right		
4	87	87	*fx*	price	reasonable		
5	41	38	*fx*	money	great value		
6	39	39	*fx*	price	worth		
7	37	37	*fx*	price	great value		
8	32	32	*fx*	money	good		
9	28	28	*fx*	price	fair		
10	28	28	*fx*	price	like		
11	28	28	*fx*	price	love		
12	26	26	*fx*	price point	excellent		
13	24	24	*fx*	investment	excellent		
14	24	24	*fx*	price too	excellent		
15	23	23	*fx*	charge	good		
16	23	23	*fx*	price	much better		
17	19	19	*fx*	money	good buy		
18	17	17	*fx*	price	good buy		
19	17	17	*fx*	price	satisfied		
20	15	15	*fx*	cost	right		
21	15	15	*fx*	value for mon	good		
22	13	13	*fx*	cost	excellent		
23	13	13	*fx*	price	meets needs		
24	13	13	*fx*	price	no problem		
25	12	12	*fx*	value for mon	excellent		
26	11	9	*fx*	pay	excellent		
27	11	11	*fx*	price	accurate		
28	10	10	*fx*	charge	excellent		
29	9	9	*fx*	cost	good		
30	9	9	*fx*	money	excellent		
31	8	8	*fx*	budget tablet	excellent		
32	7	7	*fx*	charge	fast		

1 / 1

To contrast, we looked at negative budget reviews.



473 patterns

Global	In	Type 1	Type 2	Type 3	Type 4
675	*fx*	<Unknown>	<Negative>	<Unknown>	
628	*fx*	<Unknown>	<Positive>	<Budget>	
559	*fx*	<NegativeBudget>			
522	*fx*	<Positive>	<Unknown>		
510	*fx*	<Organization>	<Positive>		
497	*fx*	<PositiveFeeling>			
465		<Weights-Measur			
426	*fx*	<Unknown>	<PositiveFeeling>		
408		<Unknown>	<Uncertain>		
393		<Unknown>	<Contextual>	<Unknown>	
376	*fx*	<Unknown>	<PositiveRecomm		
355	*fx*	<Budget>	<Negative>		
334	*fx*	<Unknown>	<NegativeFeeling>		
318		<Suggestion>			
302	*fx*	<NegativeFeeling>			

Selected: 106 patterns

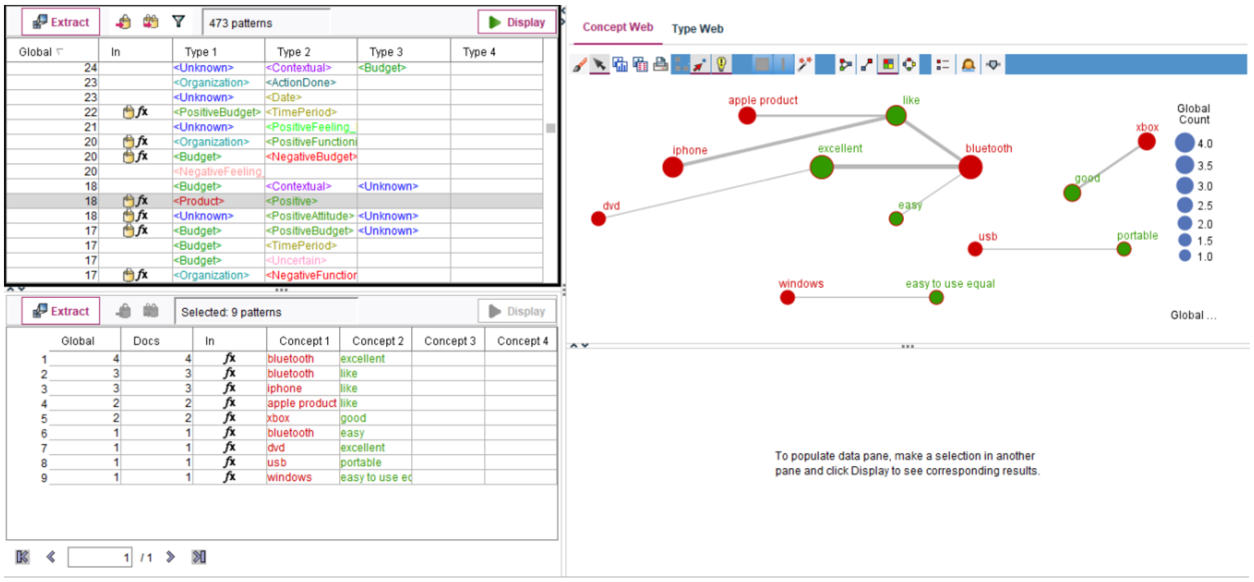
	Global	Docs	In	Concept 1	Concept 2	Concept 3	Concept 4
1	48	48	*fx*	price	bad		
2	33	33	*fx*	cost	less		
3	25	25	*fx*	price	lower		
4	16	16	*fx*	money	less		
5	15	15	*fx*	charge	long		
6	11	11	*fx*	charge	faster		
7	11	11	*fx*	charge	too long		
8	9	9	*fx*	money	not worth		
9	8	8	*fx*	price	not believable		
10	6	6	*fx*	charge	slow		
11	5	5	*fx*	money	waste		
12	5	5	*fx*	pay	remove		
13	5	5	*fx*	price tag	lower		

1 / 1

To populate data pane, make a selection in another pane and click Display to see corresponding results.

Checking positive budget reviews tell us that people are happy with the price. They find the charge and prices of products good. Checking negative reviews do not tell us much. However, there is still some negative reviews exist about the money is not worth it and price could be better.

The next step is checking what products have the negative and positive reviews and what these reviews about. By checking positive reviews of products, the result tells us that people love Apple products. The most positive reviews include excellent, like and easy keyword which related to Apple and Iphone.



473 patterns

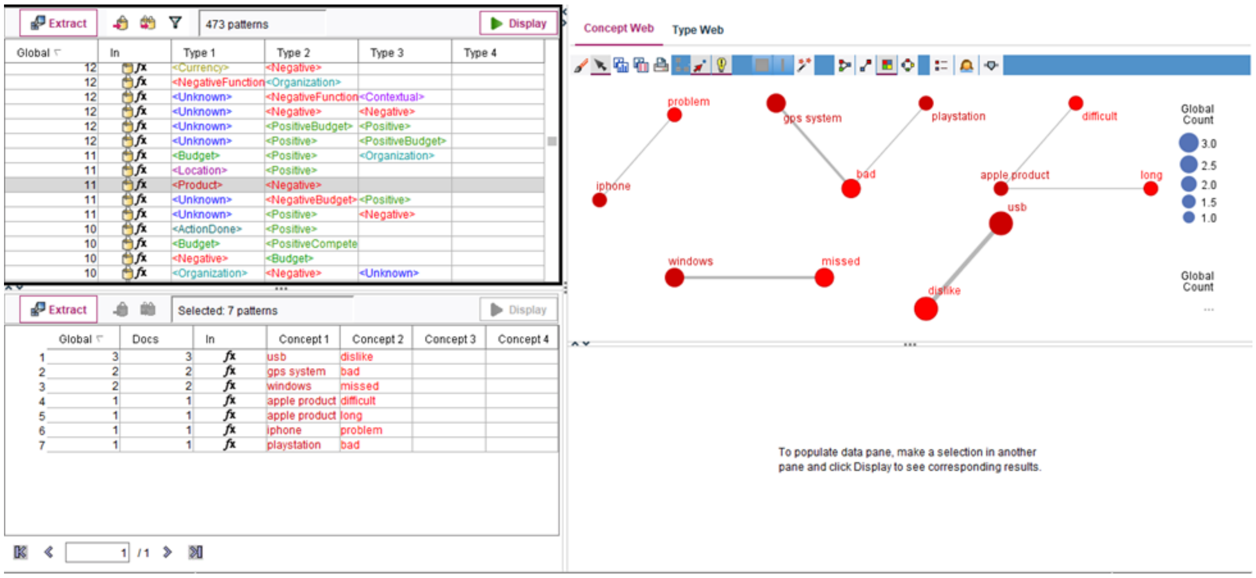
Global	In	Type 1	Type 2	Type 3	Type 4
24		<Unknown>	<Contextual>	<Budget>	
23		<Organization>	<ActionDone>		
23		<Unknown>	<Date>		
22	fx	<PositiveBudget>	<TimePeriod>		
21		<Unknown>	<PositiveFeeling_		
20	fx	<Organization>	<PositiveFunctioni		
20	fx	<Budget>	<NegativeBudget>		
20		<NegativeFeeling_			
18		<Budget>	<Contextual>	<Unknown>	
18	fx	<Product>	<Positive>		
18	fx	<Unknown>	<PositiveAttitude>	<Unknown>	
17	fx	<Budget>	<PositiveBudget>	<Unknown>	
17		<Budget>	<TimePeriod>		
17		<Budget>	<Uncertain>		
17	fx	<Organization>	<NegativeFunctior		

Selected: 9 patterns

	Global	Docs	In	Concept 1	Concept 2	Concept 3	Concept 4
1	4	4	fx	bluetooth	excellent		
2	3	3	fx	bluetooth	like		
3	3	3	fx	iphone	like		
4	2	2	fx	apple product	like		
5	2	2	fx	xbox	good		
6	1	1	fx	bluetooth	easy		
7	1	1	fx	dvd	excellent		
8	1	1	fx	usb	portable		
9	1	1	fx	windows	easy to use ec		

To populate data pane, make a selection in another pane and click Display to see corresponding results.

To contrast, checking the negative reviews of product tell us that there are almost none bad reviews and the reviews are most likely transition of one brand to another such as changing the electronic devices to Microsoft from apple and not being able to use it efficiently.



473 patterns

Global	In	Type 1	Type 2	Type 3	Type 4
12	*fx*	<Currency>	<Negative>		
12	*fx*	<NegativeFunction	<Organization>		
12	*fx*	<Unknown>	<NegativeFunction	<Contextual>	
12	*fx*	<Unknown>	<Negative>	<Negative>	
12	*fx*	<Unknown>	<PositiveBudget>	<Positive>	
12	*fx*	<Unknown>	<Positive>	<PositiveBudget>	
11	*fx*	<Budget>	<Positive>	<Organization>	
11	*fx*	<Location>	<Positive>		
11	*fx*	<Product>	<Negative>		
11	*fx*	<Unknown>	<NegativeBudget>	<Positive>	
11	*fx*	<Unknown>	<Positive>	<Negative>	
10	*fx*	<ActionDone>	<Positive>		
10	*fx*	<Budget>	<PositiveCompete		
10	*fx*	<Negative>	<Budget>		
10	*fx*	<Organization>	<Negative>	<Unknown>	

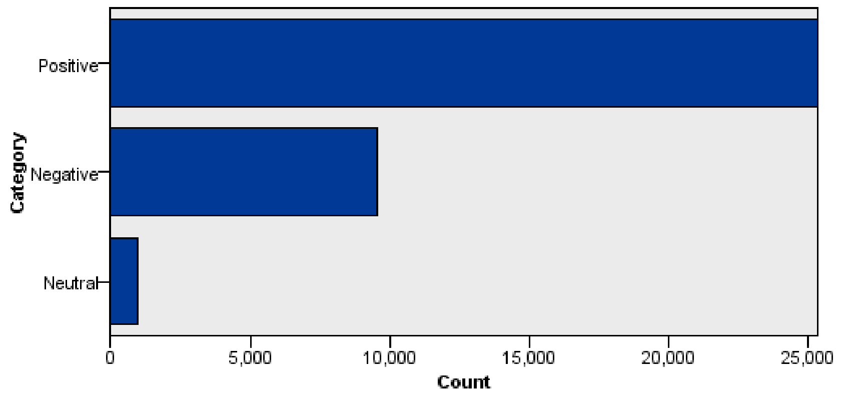
Selected: 7 patterns

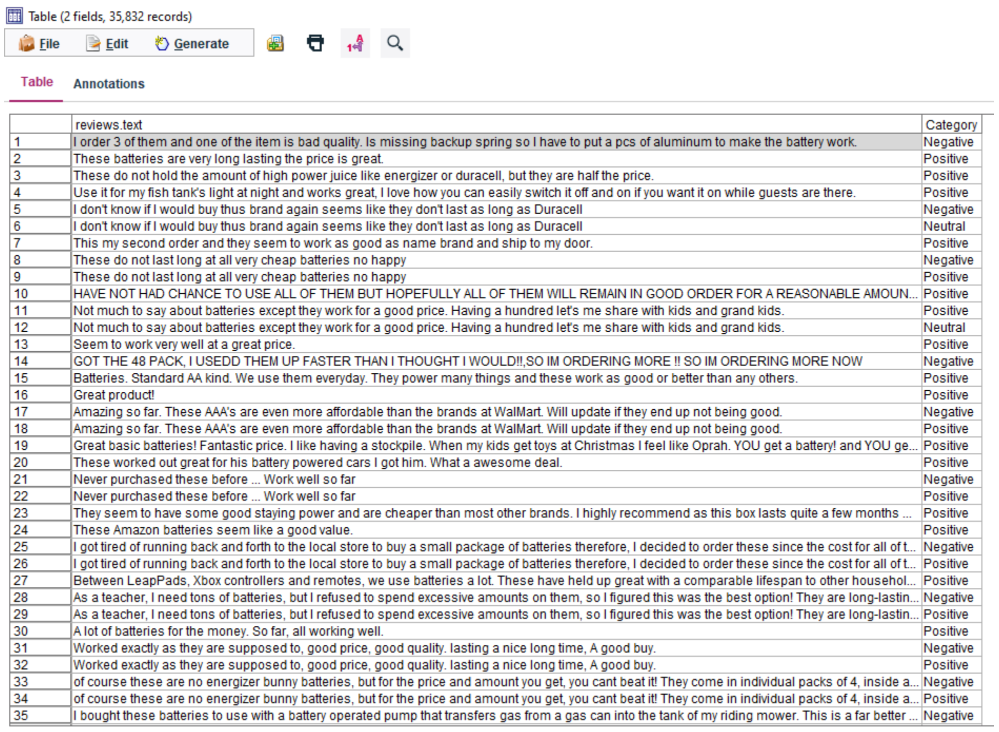
	Global	Docs	In	Concept 1	Concept 2	Concept 3	Concept 4
1	3	3	*fx*	usb	dislike		
2	2	2	*fx*	gps system	bad		
3	2	2	*fx*	windows	missed		
4	1	1	*fx*	apple product	difficult		
5	1	1	*fx*	apple product	long		
6	1	1	*fx*	iphone	problem		
7	1	1	*fx*	playstation	bad		

1 / 1

To populate data pane, make a selection in another pane and click Display to see corresponding results.

Overall scoring sentiment analysis, it gives a clear picture of the category of reviews.





	reviews.text	Category
1	I order 3 of them and one of the item is bad quality. Is missing backup spring so I have to put a pcs of aluminum to make the battery work.	Negative
2	These batteries are very long lasting the price is great.	Positive
3	These do not hold the amount of high power juice like energizer or duracell, but they are half the price.	Positive
4	Use it for my fish tank's light at night and works great, I love how you can easily switch it off and on if you want it on while guests are there.	Positive
5	I don't know if I would buy thus brand again seems like they don't last as long as Duracell	Negative
6	I don't know if I would buy thus brand again seems like they don't last as long as Duracell	Neutral
7	This my second order and they seem to work as good as name brand and ship to my door.	Positive
8	These do not last long at all very cheap batteries no happy	Negative
9	These do not last long at all very cheap batteries no happy	Positive
10	HAVE NOT HAD CHANCE TO USE ALL OF THEM BUT HOPEFULLY ALL OF THEM WILL REMAIN IN GOOD ORDER FOR A REASONABLE AMOUN...	Positive
11	Not much to say about batteries except they work for a good price. Having a hundred let's me share with kids and grand kids.	Positive
12	Not much to say about batteries except they work for a good price. Having a hundred let's me share with kids and grand kids.	Neutral
13	Seem to work very well at a great price.	Positive
14	GOT THE 48 PACK, I USEDD THEM UP FASTER THAN I THOUGHT I WOULD!!,SO IM ORDERING MORE !! SO IM ORDERING MORE NOW	Negative
15	Batteries. Standard AA kind. We use them everyday. They power many things and these work as good or better than any others.	Positive
16	Great product!	Positive
17	Amazing so far. These AAA's are even more affordable than the brands at WalMart. Will update if they end up not being good.	Negative
18	Amazing so far. These AAA's are even more affordable than the brands at WalMart. Will update if they end up not being good.	Positive
19	Great basic batteries! Fantastic price. I like having a stockpile. When my kids get toys at Christmas I feel like Oprah. YOU get a battery! and YOU ge...	Positive
20	These worked out great for his battery powered cars I got him. What a awesome deal.	Positive
21	Never purchased these before ... Work well so far	Negative
22	Never purchased these before ... Work well so far	Positive
23	They seem to have some good staying power and are cheaper than most other brands. I highly recommend as this box lasts quite a few months ...	Positive
24	These Amazon batteries seem like a good value.	Positive
25	I got tired of running back and forth to the local store to buy a small package of batteries therefore, I decided to order these since the cost for all of t...	Negative
26	I got tired of running back and forth to the local store to buy a small package of batteries therefore, I decided to order these since the cost for all of t...	Positive
27	Between LeapPads, Xbox controllers and remotes, we use batteries a lot. These have held up great with a comparable lifespan to other househol...	Positive
28	As a teacher, I need tons of batteries, but I refused to spend excessive amounts on them, so I figured this was the best option! They are long-lastin...	Negative
29	As a teacher, I need tons of batteries, but I refused to spend excessive amounts on them, so I figured this was the best option! They are long-lastin...	Positive
30	A lot of batteries for the money. So far, all working well.	Positive
31	Worked exactly as they are supposed to, good price, good quality. lasting a nice long time, A good buy.	Negative
32	Worked exactly as they are supposed to, good price, good quality. lasting a nice long time, A good buy.	Positive
33	of course these are no energizer bunny batteries, but for the price and amount you get, you cant beat it! They come in individual packs of 4, inside a...	Negative
34	of course these are no energizer bunny batteries, but for the price and amount you get, you cant beat it! They come in individual packs of 4, inside a...	Positive
35	I bought these batteries to use with a battery operated pump that transfers gas from a gas can into the tank of my riding mower. This is a far better ...	Negative

Sentiment Analysis (Reena)

Data Cleaning & Pre-processing: (Reena)

In this section, data cleaning and pre-processing of the dataset will be explained in detail. To begin with, data set "1429_1.csv" was uploaded into a dataframe. After uploading the dataset, we checked the dimensionality as well as the columns of our data frame. We found 34660 rows and 21 columns out of which many columns had missing values as shown in the figure below (Figure 15):

Note: The python code ran to arrive results in this section is attached in the appendix section.

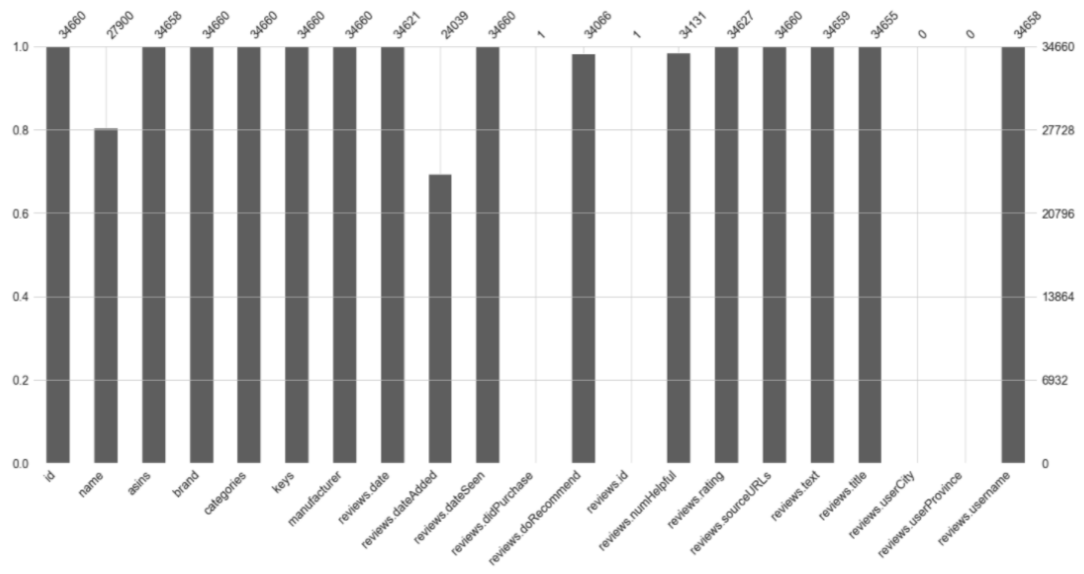


Figure 15

The above figure shows that some of the columns such as name, reviews.dateAdded, reviews.didPurchase, reviews.id, reviews.userCity, reviews.userProvince have total number of missing values above 6000 each. Since we will be dealing with only three columns for our analysis such as id, reviews.text, reviews.rating, our primary focus is to find out missing values in these three columns.

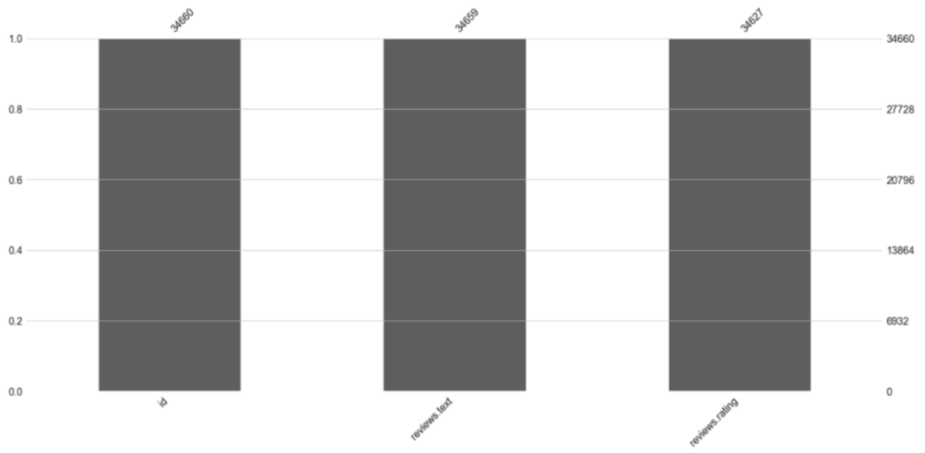
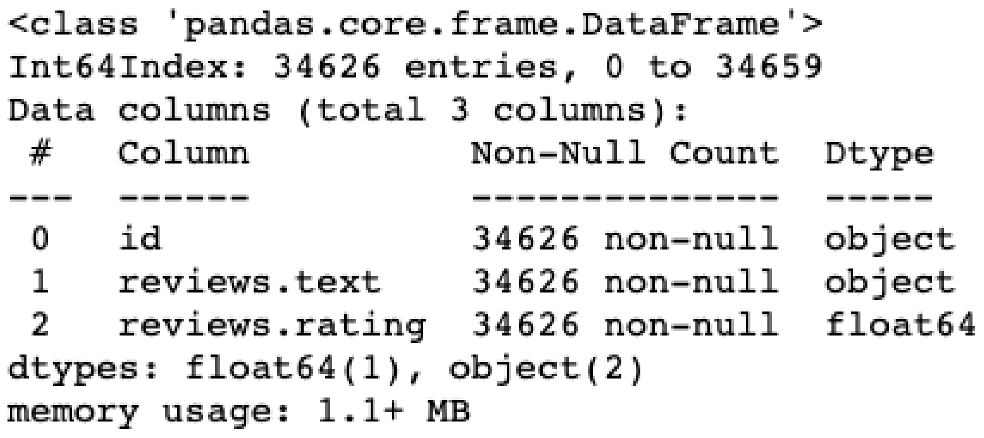


Figure 16

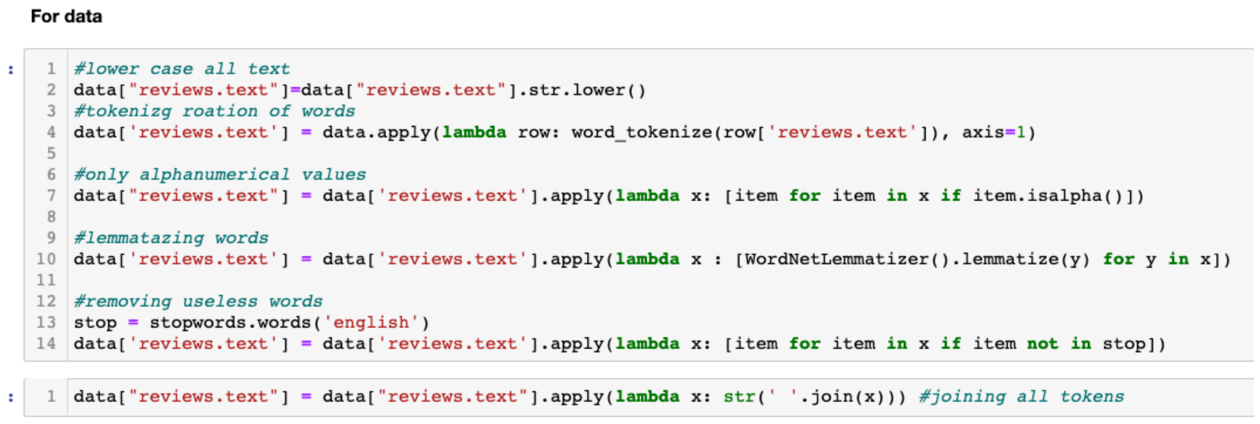
As we can see in the above (Figure 16) that there is one value missing in column reviews.text whereas there are 32 missing values in the column reviews.rating. Since, we have sufficient number of instances, we're going to remove all those instances with missing values instead of replacing NaN with the mean of the others. After removing, we were left with following number of rows and columns as shown in figure below (Figure 17):



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 34626 entries, 0 to 34659
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   id              34626 non-null  object 
 1   reviews.text    34626 non-null  object 
 2   reviews.rating  34626 non-null  float64
dtypes: float64(1), object(2)
memory usage: 1.1+ MB
```

Figure 17

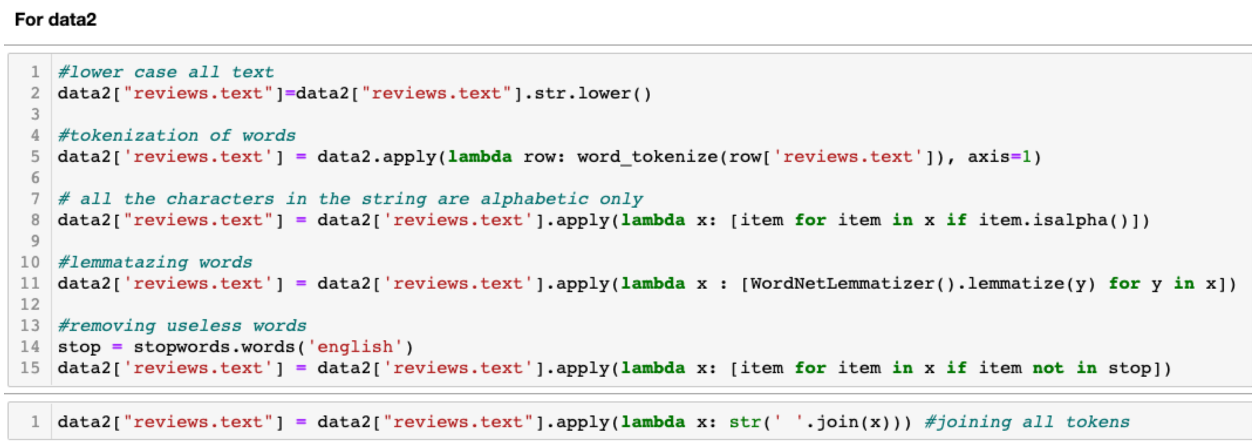
Since our dataset “1429_1.csv” was highly biased towards high ratings: 5, we uploaded additional two datasets:
Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_May19.csv,
Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv with ratings below 3.
Fortunately, we didn’t find any missing values into these two datasets.
After creating a three different dataframes: data, data1, data2 by copying columns id, reviews.text, reviews.rating from original dataframes df, df1, df2 and removing missing rows from data, we went ahead to pre-process “reviews.text” for each dataframes: data, data1, data2 separately as shown in figures below (Figure 18):



For data

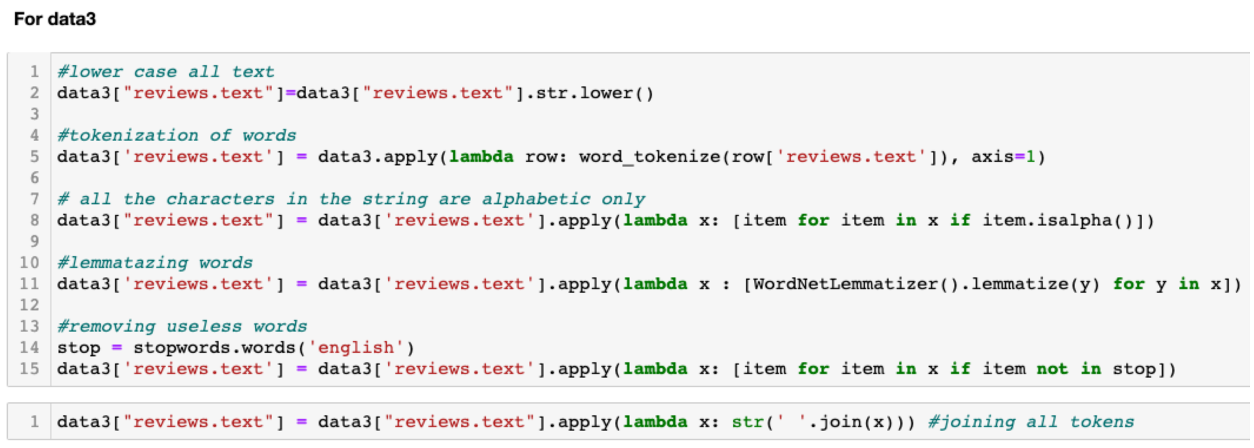
```
#lower case all text
data["reviews.text"]=data["reviews.text"].str.lower()
#tokenizg roation of words
data['reviews.text'] = data.apply(lambda row: word_tokenize(row['reviews.text']), axis=1)

#only alphanumerical values
data["reviews.text"] = data['reviews.text'].apply(lambda x: [item for item in x if item.isalpha()])

#lemmatazing words
data['reviews.text'] = data['reviews.text'].apply(lambda x : [WordNetLemmatizer().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
data['reviews.text'] = data['reviews.text'].apply(lambda x: [item for item in x if item not in stop])
```

```
data["reviews.text"] = data["reviews.text"].apply(lambda x: str(' '.join(x))) #joining all tokens
```


For data2

```
#lower case all text
data2["reviews.text"]=data2["reviews.text"].str.lower()

#tokenization of words
data2['reviews.text'] = data2.apply(lambda row: word_tokenize(row['reviews.text']), axis=1)

# all the characters in the string are alphabetic only
data2["reviews.text"] = data2['reviews.text'].apply(lambda x: [item for item in x if item.isalpha()])

#lemmatazing words
data2['reviews.text'] = data2['reviews.text'].apply(lambda x : [WordNetLemmatizer().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
data2['reviews.text'] = data2['reviews.text'].apply(lambda x: [item for item in x if item not in stop])
```

```
data2["reviews.text"] = data2["reviews.text"].apply(lambda x: str(' '.join(x))) #joining all tokens
```


For data3

```
#lower case all text
data3["reviews.text"]=data3["reviews.text"].str.lower()

#tokenization of words
data3['reviews.text'] = data3.apply(lambda row: word_tokenize(row['reviews.text']), axis=1)

# all the characters in the string are alphabetic only
data3["reviews.text"] = data3['reviews.text'].apply(lambda x: [item for item in x if item.isalpha()])

#lemmatazing words
data3['reviews.text'] = data3['reviews.text'].apply(lambda x : [WordNetLemmatizer().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
data3['reviews.text'] = data3['reviews.text'].apply(lambda x: [item for item in x if item not in stop])
```

```
data3["reviews.text"] = data3["reviews.text"].apply(lambda x: str(' '.join(x))) #joining all tokens
```

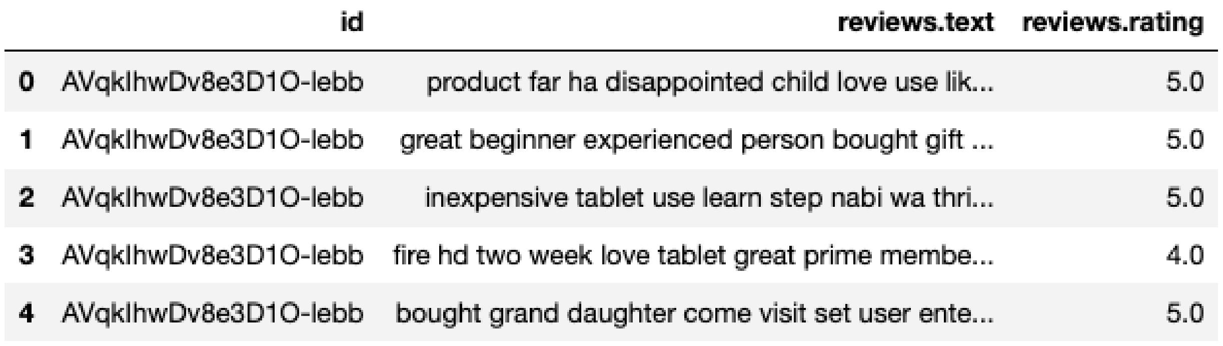
Figure 18

In order to pre-process our dataframes, we installed python library nltk, a leading platform to deal human text language. It comes with easy-to-use interfaces for text processing libraries such as classification, tokenization, stemming, tagging, parsing, and semantic reasoning and many more.

Pre-processing text begins with cleaning up irregularities by lowering the case of all text in reviews.text column. Then applied tokenization method to that column to split each sentence into list of words. Next applied lambda function to make sure only the alphabetic are included in all the characters in the string. In order to normalize list of tokenized words, WordNetLemmatizer is called. Then stopwords method was provoked to eliminate the most common words in a language such as “the”, “a”, “an”, “in”. After removing all the stopwords, all the token words were joined.

Note: The above pre-processing method applied to the three dataframes used to build model.

After cleaning and pre-processing, we created a final dataframe “finalData” by concatenating data, data1, data2. This final dataframe was finally used for running classification algorithms that will be discussed in detail in ***Analysis Model*** section. The first five rows of our new dataframe “finalData” are shown in the table below (Figure 19):



	id	reviews.text	reviews.rating
0	AVqkIhwDv8e3D1O-lebb	product far ha disappointed child love use lik...	5.0
1	AVqkIhwDv8e3D1O-lebb	great beginner experienced person bought gift ...	5.0
2	AVqkIhwDv8e3D1O-lebb	inexpensive tablet use learn step nabi wa thri...	5.0
3	AVqkIhwDv8e3D1O-lebb	fire hd two week love tablet great prime membe...	4.0
4	AVqkIhwDv8e3D1O-lebb	bought grand daughter come visit set user ente...	5.0

Figure 19

Analysis Model (Reena)

Sentiment analysis is the analysis of a piece of writing to identify positive, neutral or negative sentiment. It is a powerful tool to analyze text through the combination of natural language processing (NLP) and machine learning techniques (“Sentiment Analysis Explained”, n.d). Product reviews and ratings are a crucial part for any e-commerce businesses to learn about customer’s perspective for their products selling on the internet. Customer’s reviews and ratings are the direct reflection of positive, negative or neutral sentiments.

This paper will attempt to apply sentiment analysis using classification algorithms such as logistic regression and random forest on reviews and ratings of Amazon products. The overall aim for implementing sentiment analysis based on ratings and reviews is to build predictive model that can classify positive, negative or neutral sentiments with high precision and accuracy on a new dataset or new instances.

The analysis begins with uploading data “1429_1.csv” into data frame: df in Jupyter notebook kernel. Before uploading the dataset, necessary libraries were imported for data processing/manipulation, visualization and natural language processing. The names of the libraries are mentioned in the appendix section.

After uploading the dataset into data frame “df”, data cleaning and pre-processing were done and discussed in ***Data Cleaning & Pre-processing*** section. Once cleaning and pre-processing were done on our first dataframe “data”, we carried out Exploratory Data Analysis on the same.

Exploratory Data Analysis (Reena)

We begin exploratory data analysis by first creating a word cloud using column “reviews.text” as shown in figure below *(Figure 20):

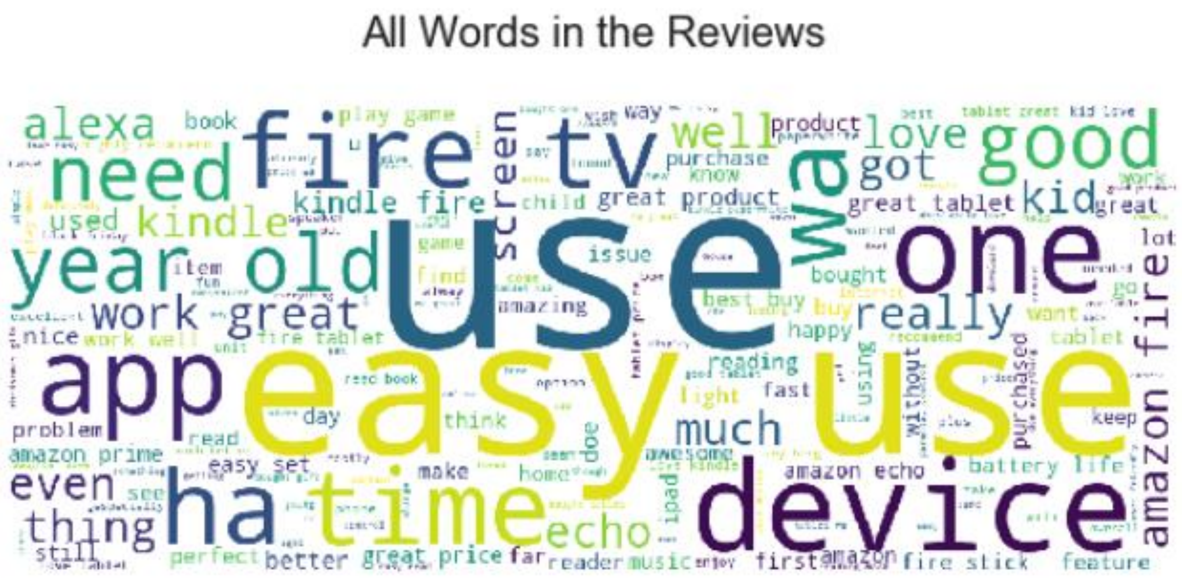


Figure 20

Words in the above results doesn't convey sentiments embedded in comments. So, we performed a word count of certain words that could describe the sentiment in a better way. In order to create a word cloud of sentimental words from the reviews.text column, we ran the algorithm with the list “words” of certain sentimental words such as 'awesome', 'great', 'fantastic', 'awful', 'hate' and so on. Then mapped those words included in a variable “words” with the entire text in reviews.text column and generated a dictionary called “**diz**” with each element in the list used most frequently in the text column. We then generated a word cloud of dictionary “**diz**” as shown in the figure below (Figure 21):

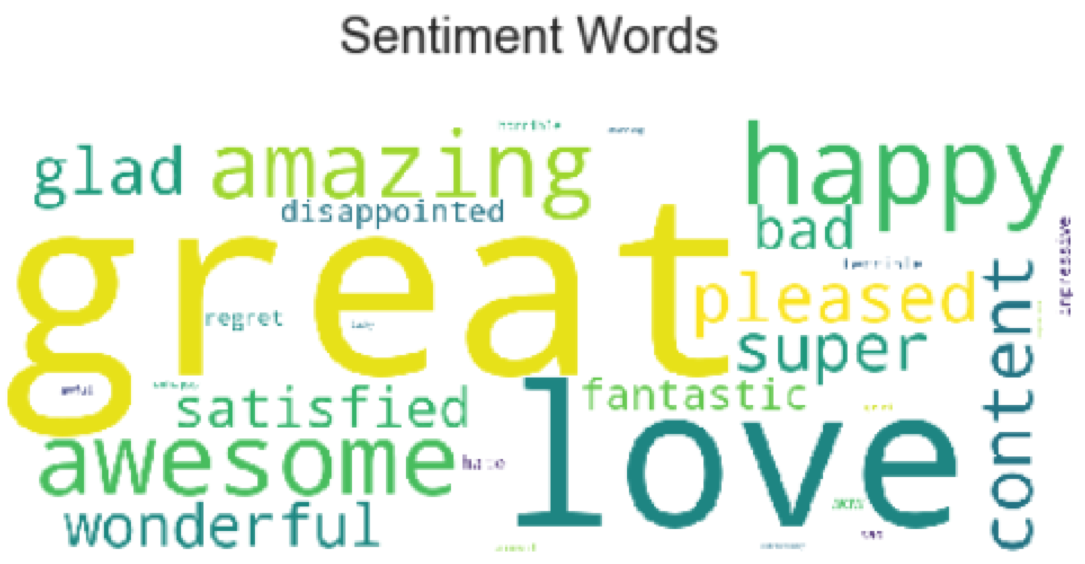
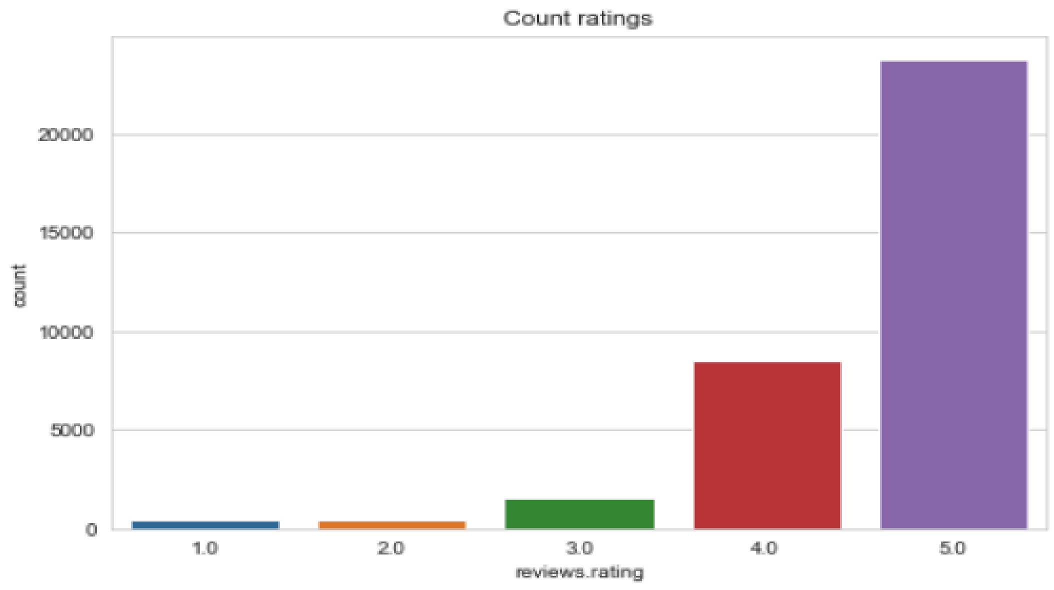
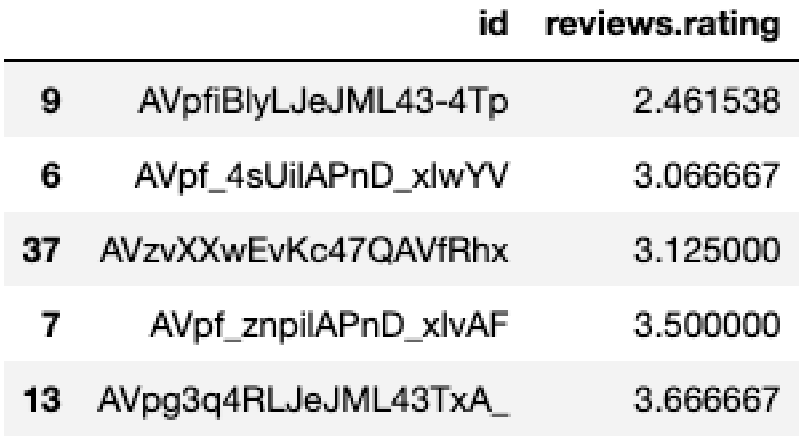


Figure 21

The result of the above wordcloud suggests that most of the comments are positive. Now, we would look at the total count each of the reviews.rating to see if there's any biased towards higher ratings. To plot total count of each rating from 1 to 5, we used seaborn countplot function.



The above count result shows majority of the ratings is 5. This suggests that data is highly biased towards high ratings. In order to confirm the presence of biasness in our dataset, we dig further to get the average rating for every product Amazon sold. Since many instances of product name column in our original dataset were missing so we decided to work with product Id instead to find out average ratings per product. We found average ratings per product through grouping reviews.rating column by “id” and applied mean function on ratings and assigned to a variable “data1” after sorting values in an ascending order as shown in the table below(Figure 22):



	id	reviews.rating
9	AVpfiBlyLJeJML43-4Tp	2.461538
6	AVpf_4sUilAPnD_xlwYV	3.066667
37	AVzvXXwEvKc47QAVfRhx	3.125000
7	AVpf_znpilAPnD_xlvAF	3.500000
13	AVpg3q4RLJeJML43TxA_	3.666667

Figure 22

The above table shows the first five rows of variable data1 that consist of product id and corresponding average ratings included in “reviews.rating” column. The barplot of reviews.rating is shown below(Figure 23):

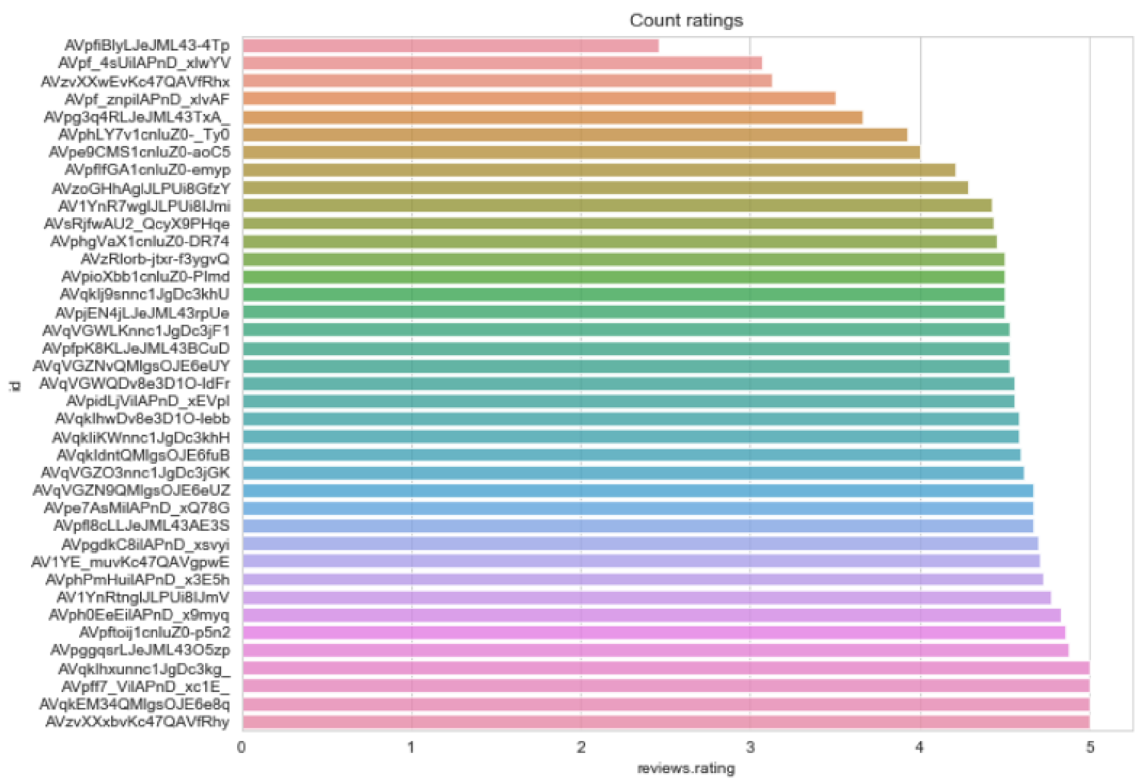


Figure 23

As we can clearly see that the reviews are heavily biased towards positive reviews and higher ratings: 4 and 5. In order to deal with this biasness of data, we decided to add the other two csv files such as Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_May19.csv, Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv and get only the reviews that have ratings lower or equal to 3. After uploading these two csv files, we copied three columns and assigned to new dataframes data2, data3 with reviews.ratings <=3 as shown below(Figure 24):

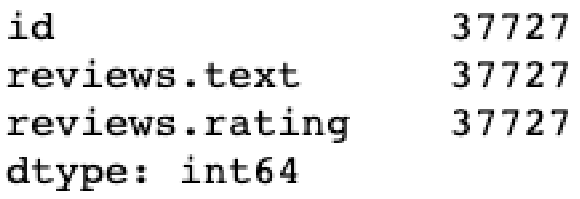
```
# load dataset
df2 = pd.read_csv('Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_May19.csv')
df3 = pd.read_csv('Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv')
```

```
data2 = df2[["id","reviews.text","reviews.rating"]]
data3 = df3[["id","reviews.text","reviews.rating"]]
```

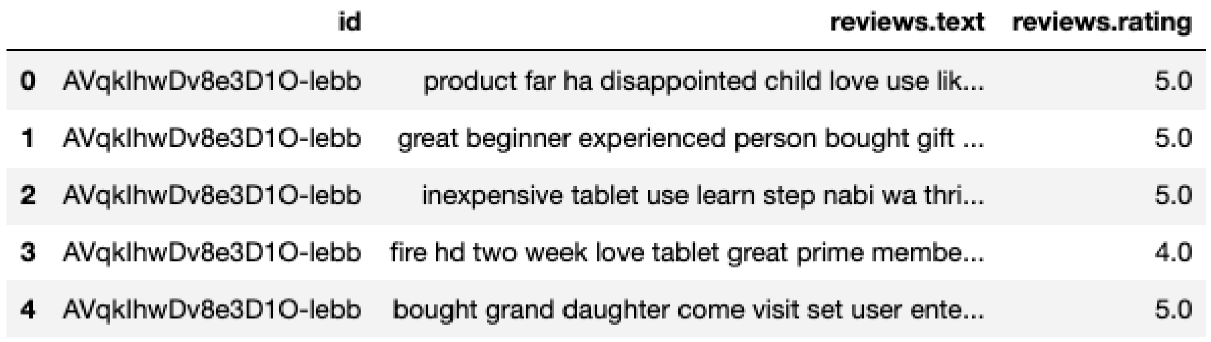
```
data2 = data2[data2["reviews.rating"]<=3]
data3 = data3[data3["reviews.rating"]<=3]
```

Figure 24

In order to carry out classification algorithms- Logistic regression and Random forest model, we concatenated three dataframes data, data2, data3. The total count of rows and first five rows of our final dataframes are shown in the tables below (Figure 25):



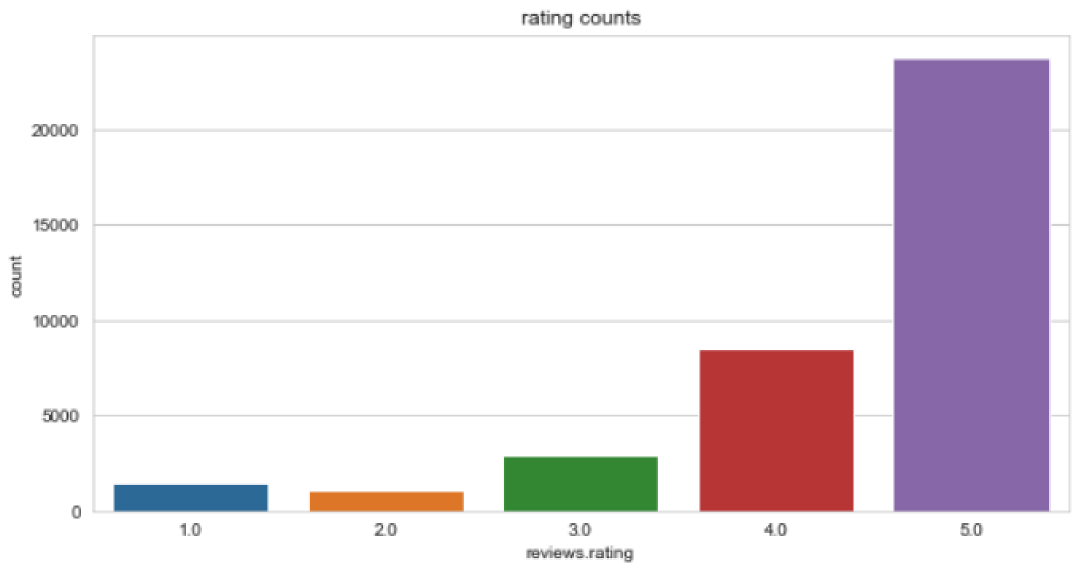
```
id                37727
reviews.text      37727
reviews.rating    37727
dtype: int64
```



	id	reviews.text	reviews.rating
0	AVqklhwDv8e3D1O-lebb	product far ha disappointed child love use lik...	5.0
1	AVqklhwDv8e3D1O-lebb	great beginner experienced person bought gift ...	5.0
2	AVqklhwDv8e3D1O-lebb	inexpensive tablet use learn step nabi wa thri...	5.0
3	AVqklhwDv8e3D1O-lebb	fire hd two week love tablet great prime membe...	4.0
4	AVqklhwDv8e3D1O-lebb	bought grand daughter come visit set user ente...	5.0

Figure 25

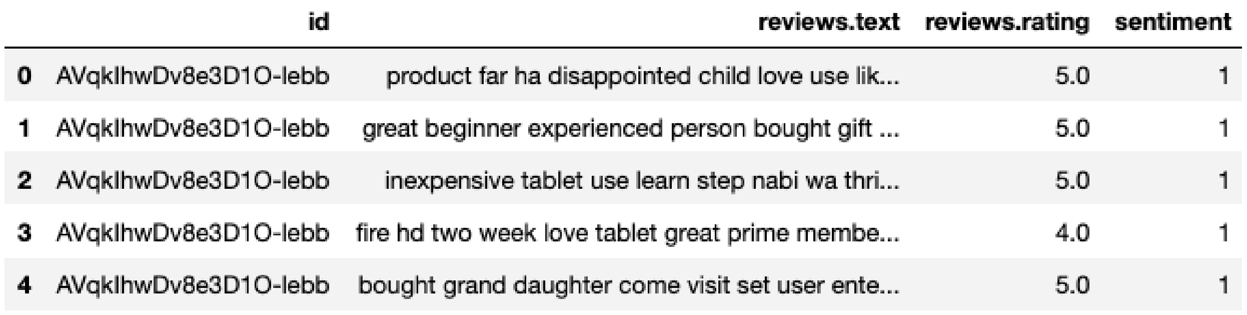
We again plotted the count plot of review.rating column to see if we have been successful in removing biasness in ratings.



Unfortunately, we are still seeing that rating '5' is far ahead of other lower ratings in the above figure. Still, there's biasness in our dataset.

Predictive Modeling using ML algorithms (Reena)

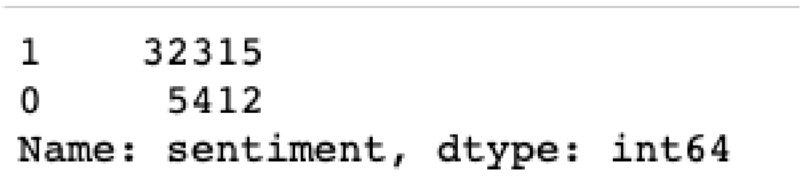
We would see that after training the model if there's any differences in the results. Next, we assigned sentiments to all the ratings i.e. we assigned positive sentiment (1) for the ratings are 4 and 5 and assigned a negative sentiment (0) for the ratings are 1,2,3. Later, we mapped these sentiments based on reviews.rating to our dataframe: finalData. The first five rows post adding sentiment column into our dataframe finalData is shown below (Figure 26):



	id	reviews.text	reviews.rating	sentiment
0	AVqkIhwDv8e3D1O-lebb	product far ha disappointed child love use lik...	5.0	1
1	AVqkIhwDv8e3D1O-lebb	great beginner experienced person bought gift ...	5.0	1
2	AVqkIhwDv8e3D1O-lebb	inexpensive tablet use learn step nabi wa thri...	5.0	1
3	AVqkIhwDv8e3D1O-lebb	fire hd two week love tablet great prime membe...	4.0	1
4	AVqkIhwDv8e3D1O-lebb	bought grand daughter come visit set user ente...	5.0	1

Figure 26

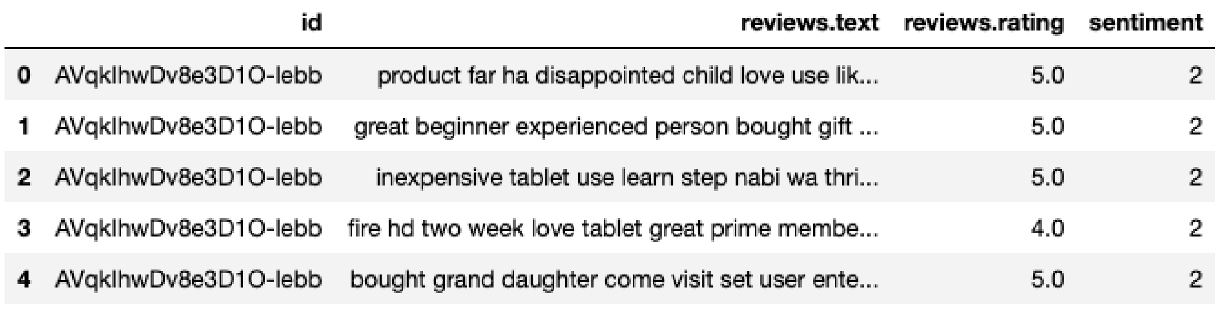
Then we counted the number of positive and negative sentiments in the sentiment column.



```
1    32315
0     5412
Name: sentiment, dtype: int64
```

The above result shows that our data is still heavily biased towards positive sentiments. Then we built the tfidf matrix to train models by importing TfidfVectorizer from sklearn’s feature_extraction.text family. After transforming "reviews.text" column through vectorization and assigned to a variable “text”, we imported train_test_split, and classification_report from sklearn.model_selection and sklearn.metrics family. We then split our dataset into test and train subsets by keeping test size 0.3 where x_train and x_test included text element and y_train, and y_test included finalData["sentiment"] element.

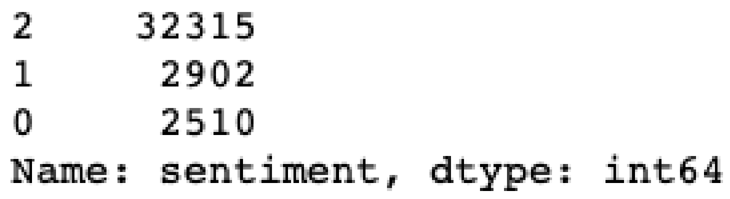
We first ran logistic regression model, and later random forest. The results of both the models are in the output section of the paper. Overall, the performances were satisfactory but then we added a third label "neutral sentiment" value in the sentiment column for better results. We then mapped the sentiments i.e., positive sentiment, 1: neutral sentiment 0: negative sentiment corresponding to reviews. rating column. The first five rows of the dataframe after adding three labels are shown below (Figure 27):



	id	reviews.text	reviews.rating	sentiment
0	AVqkIhwDv8e3D1O-lebb	product far ha disappointed child love use lik...	5.0	2
1	AVqkIhwDv8e3D1O-lebb	great beginner experienced person bought gift ...	5.0	2
2	AVqkIhwDv8e3D1O-lebb	inexpensive tablet use learn step nabi wa thri...	5.0	2
3	AVqkIhwDv8e3D1O-lebb	fire hd two week love tablet great prime membe...	4.0	2
4	AVqkIhwDv8e3D1O-lebb	bought grand daughter come visit set user ente...	5.0	2

Figure 27

The count of each three sentiment is shown below (Figure 28):



```
2    32315
1     2902
0     2510
Name: sentiment, dtype: int64
```

Figure 28

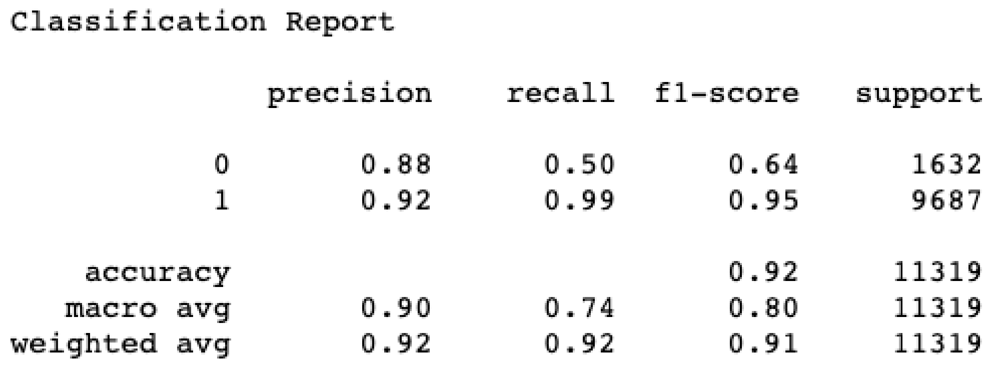
Still, we can see in the above table that our data is heavily biased towards the positive sentiment. We again created a matrix of text of reviews.text column through vector transformation method. We again split our dataset into test and train subsets by keeping test size 0.3 where x_train and x_test included text element and y_train, and y_test included finalData["sentiment"] element.

Later we first ran logistic regression model, and random forest. The results and interpretation of the results of both the models are in the output, interpretation sections of our paper.

Model Results (Reena)

- *Results of Logistic regression with two labels: positive and negative:*

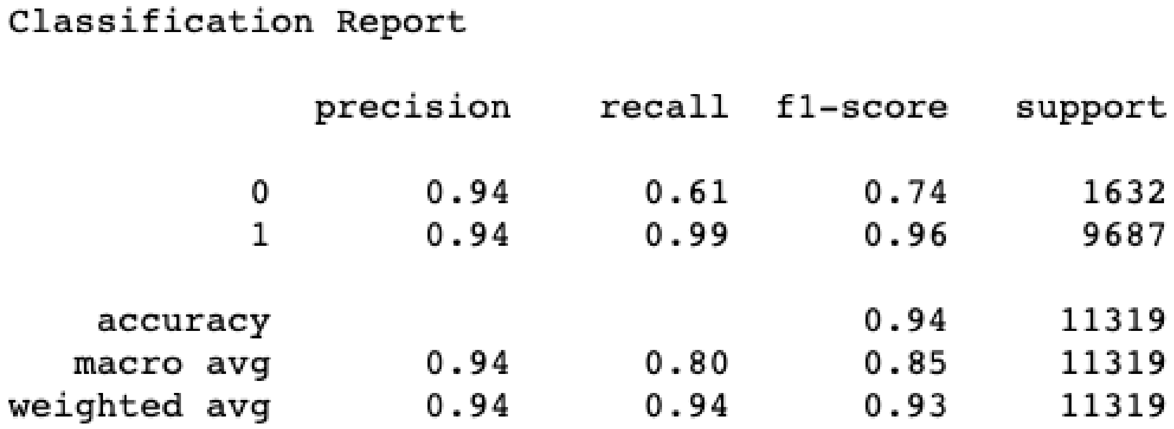
  Test accuracy 0.9055



```
Classification Report

              precision    recall  f1-score   support

           0       0.88      0.50      0.64      1632
           1       0.92      0.99      0.95      9687

    accuracy                           0.92     11319
   macro avg       0.90      0.74      0.80     11319
weighted avg       0.92      0.92      0.91     11319
```

- *Results of Random Forest with two labels: positive and negative*:

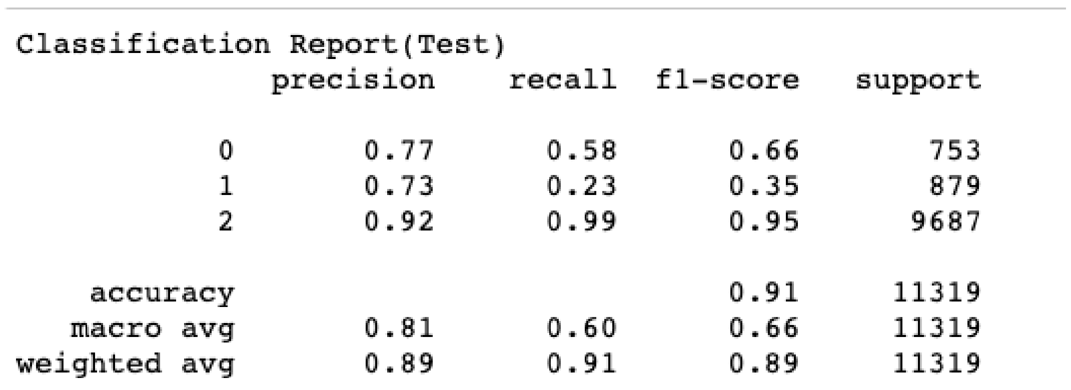
```
Test accuracy 0.9274
```



```
Classification Report

              precision    recall  f1-score   support

           0       0.94      0.61      0.74      1632
           1       0.94      0.99      0.96      9687

    accuracy                           0.94     11319
   macro avg       0.94      0.80      0.85     11319
weighted avg       0.94      0.94      0.93     11319
```

- *Results of Logistic regression with three labels: positive, neutral and negative:*

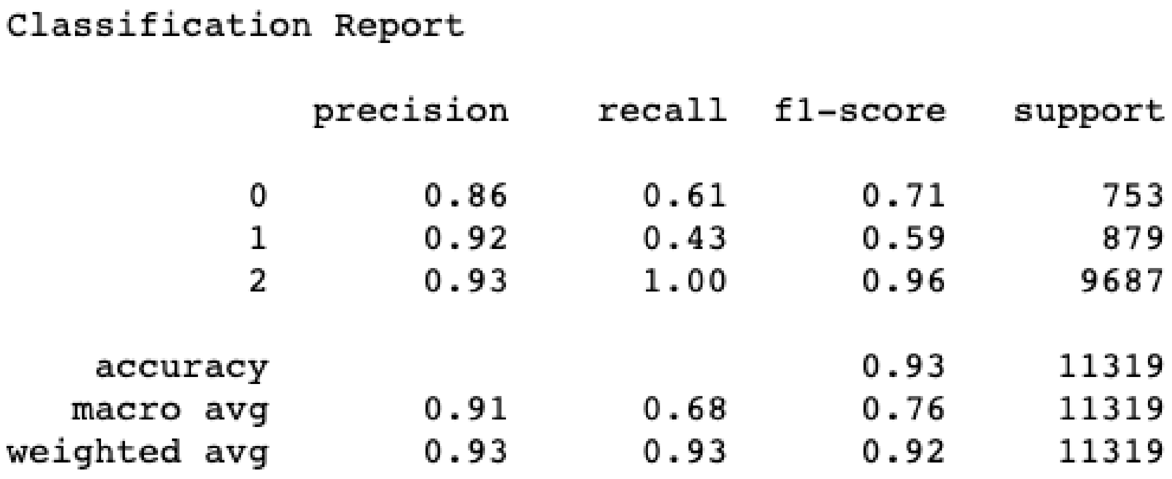
```
Test accuracy 0.9055
```



```
Classification Report(Test)
              precision    recall  f1-score   support

           0       0.77      0.58      0.66       753
           1       0.73      0.23      0.35       879
           2       0.92      0.99      0.95      9687

    accuracy                           0.91     11319
   macro avg       0.81      0.60      0.66     11319
weighted avg       0.89      0.91      0.89     11319
```

- *Results of Random Forest with three labels: positive, neutral and negative:*

```
Test accuracy 0.9274
```



```
Classification Report

              precision    recall  f1-score   support

           0       0.86      0.61      0.71       753
           1       0.92      0.43      0.59       879
           2       0.93      1.00      0.96      9687

    accuracy                           0.93     11319
   macro avg       0.91      0.68      0.76     11319
weighted avg       0.93      0.93      0.92     11319
```

Model Interpretation (Reena)

- **With Label 2:**
  - The precision rate (0.94) for both negative and posistive sentiments is higher and performed better than Logistic regression. Even if we compare the recall and f1-score of negative sentiments, we will find that the Random Forests model (recall:0.61, f1-score:0.74) has performed better again than Logistic regression (recall:0.50, f1-score:0.64). In terms of accuracy rate, Random Forests model (0.9274) has performed better than Logistic Regression (0.9055).

- **With Label 3:**
    - Accuracy score: Accuracy score of Random forest (0.9274) is higher than that of the logistic regression (0.9055). Precision scores for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression. Recall rates for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression. F1-score for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression

Final Conclusion

Utilizing the IBM SPSS Modeler, we were able to determine that there was generally more positive sentiment within the reviews then there was negative. Products with lower reviews are not very significant enough to predict the reason and the products with higher reviews are mostly superior that performs well in all aspects. It is expected that these products will keep selling at high level. Overall, the products that have been purchased and left a review on Amazon meet the expectation and satisfies the customers. This was validated by sentiment analysis in Python.

If we compare the performance of Logistic regression with that of the random forest after adding third label neutral sentiment, it is pretty apparently that based on:

- **Accuracy score:** Accuracy score of Random forest (0.9274) is higher than that of the logistic regression (0.9055)
- **Precision scores** for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression
- **Recall rates** for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression
- **F1-score** for all three sentiments- negative:0, nuetral:1, positive: 2 are better in Random Forest model than logistic regression.

Through model evaluation, we have decided to go with the latest **Random Forest model** with label three as it provides better results.

Final Recommendation

Based on our analysis, we have tried to achieve the best possible score. However, the results could have been improved by:

- Balancing the dataset either through increasing the number of negative and neutral labels or by decreasing the positive labels.
- We could have applied 5-fold cross validation for parameter tuning using GridSearchCV method to arrive at optimal number of estimators and criteria either gini or entropy for splitting at features that attain least impurity for classification
- We could have run ensembles of classification models to improve the overall performance of the model.

Overall, we’re satisfied that we were able to achieve desired results of model performances and understand thoroughly what people say good or bad about price and products that are being sold by Amazon on its own website.

Reference

Calvi, M. (2019, July 08). Amazon Reviews Sentiment Analysis. Retrieved December 15, 2020, from https://www.kaggle.com/lele1995/amazon-reviews-sentiment-analysis

Chavan, J. (2020, May 08). NLP: Tokenization, Stemming, Lemmatization, Bag of Words, TF-IDF, POS. Retrieved December 16, 2020, from https://medium.com/@jeevanchavan143/nlp-tokenization-stemming-lemmatization-bag-of-words-tf-idf-pos-7650f83c60be

Ghelber, A. (2020, March 04). Amazon Review Analysis: The Beginners Guide (2020) [Digital image]. Retrieved December 18, 2020, from https://www.revuze.it/blog/amazon-review-analysis/

Jayaprakash, A. (2020, November 1). Building a sentiment classification model [Digital image]. Retrieved December 15, 2020, from https://abinaya-j.medium.com/building-a-sentiment-classification-model-a2b5fe24e9a1

Sentiment Analysis Explained. (n.d.). Retrieved December 15, 2020, from https://www.lexalytics.com/technology/sentiment-analysis

Sentiment Analysis of Tweets. (2016, April 22). Retrieved December 16, 2020, from https://induchandra.wordpress.com/contact/

Zhang, Mick (2020). Amazon Reviews Using Sentiment Analysis. Retrieved December 18, 2020, from https://mickzhang.com/amazon-reviews-using-sentiment-analysis/

Appendix

```
#import required libraries
import pandas as pd # data processing/manipulation
import numpy as np # linear algebra
# Visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```
#!pip install wordcloud
```

In [3]:

```
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
from wordcloud import WordCloud
import nltk
nltk.download('wordnet')
[nltk_data] Downloading package wordnet to
[nltk_data]     /Users/reenasehitya/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
#pip install missingno
```

In [94]:

```
import missingno as msno
```

In [6]:

```
nltk.download('stopwords')
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/reenasehitya/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
import os
import warnings
```

In [8]:

```
os.getcwd()
```

Out[8]:

```
'/Users/reenasehitya/GGU/MSBA327/Final Project'
```

In [9]:

```
# load dataset
df = pd.read_csv('1429_1.csv')
df.head()
```

```
# To check the the dimensionality of our DataFrame
df.shape # there are 34660 rows and 21 columns
```

```
# To understand all the column available in the dataset
df.info()
```

```
#Checking missing values.
```

```
df.isna().sum() # there are various columns with missing values, specially
column 'name',
```

```
sns.set_style('whitegrid')
```

In [105]:

```
# Visualize missing values as a matrix
msno.bar(df)
plt.show()
```

```
data = df[['id','reviews.text', 'reviews.rating']]
# I have selected column 'id' instaed of column 'name' because 'name' column
has missing values
```

In [103]:

```
data.head()
```

```
# To describe on object data types
data.describe(include=["O"])
```

```
import missingno as msno
msno.bar(data)
```

```
data.info()
```

```
data = data.dropna()
data.info() # now we will be working on 34626 instances.
```

For data

In [21]:

```
#lower case all text
data["reviews.text"]=data["reviews.text"].str.lower()
#tokenizg roation of words
data['reviews.text'] = data.apply(lambda row: word_tokenize(row['reviews.text
']), axis=1)

#only alphanumerical values
data["reviews.text"] = data['reviews.text'].apply(lambda x: [item for item in
x if item.isalpha()])

#lemmatazing words
data['reviews.text'] = data['reviews.text'].apply(lambda x : [WordNetLemmatiz
er().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
```

```
data['reviews.text'] = data['reviews.text'].apply(lambda x: [item for item in
x if item not in stop])
```
In [22]:
```
data["reviews.text"] = data["reviews.text"].apply(lambda x: str(' '.join(x)))
#joining all tokens
```
In [23]:
```
data.head()
```

Exploratory Data Analysis

In [24]:
```
rt = data['reviews.text']
wordcloud = WordCloud(background_color='white',
                      width=1000,
                      height=400
                     ).generate(" ".join(rt))
plt.figure(figsize=(10,5))
plt.imshow(wordcloud)
plt.title('All Words in the Reviews\n',size=20)
plt.axis('off')
plt.show()
```

```
words = ['awesome','great','fantastic','extraordinary','amazing','super',
                 'magnificent','stunning','impressive','wonderful','breathtak
ing',
                 'love','content','pleased','happy','glad','satisfied','lucky
',
                 'shocking','cheerful','wow','sad','unhappy','horrible','regr
et',
                 'bad','terrible','annoyed','disappointed','upset','awful','h
ate']
```
In [26]:
```
data['reviews.text']

rt = " ".join(data['reviews.text'])
```
In [28]:
```
diz = {}
for word in rt.split(" "):
    if word in words:
        diz[word] = diz.get(word,0)+1
```
In [29]:
```
wordcloud = WordCloud(background_color='white',
                      width=1000,
                      height=400
                     ).generate_from_frequencies(diz)
plt.figure(figsize=(8,5))
plt.imshow(wordcloud)
plt.title('Sentiment Words\n',size=20)
plt.axis('off')
plt.show()

plt.figure(figsize=(8,5))
sns.countplot(data['reviews.rating'])
plt.title('Count ratings')
plt.show()
```

```
data.head()
```

```
data1 = data.groupby("id")['reviews.rating'].mean().reset_index()
```

In [33]:

```
data1.head()
```

```
data1 = data1.sort_values(['reviews.rating'])
data1.head()
```

```
plt.figure(figsize=(10,8))
sns.barplot(x=data1["reviews.rating"], y=data1["id"], alpha = 0.8)
plt.title('Count ratings')
plt.show()
```

```
# load dataset
df2 = pd.read_csv('Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products_M
ay19.csv')
df3 = pd.read_csv('Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.c
sv')
```

```
data2 = df2[["id","reviews.text","reviews.rating"]]
data3 = df3[["id","reviews.text","reviews.rating"]]
```

In [39]:

```
data2 = data2[data2["reviews.rating"]<=3]
data3 = data3[data3["reviews.rating"]<=3]
```

In [106]:

```
len(data2), len(data3)
```

```
data2.isna().sum()
```

```
data2.isna().sum()
```

For data2

In [41]:

```
#lower case all text
data2["reviews.text"]=data2["reviews.text"].str.lower()
```

```
#tokenization of words
data2['reviews.text'] = data2.apply(lambda row: word_tokenize(row['reviews.te
xt']), axis=1)

# all the characters in the string are alphabetic only
data2["reviews.text"] = data2['reviews.text'].apply(lambda x: [item for item
in x if item.isalpha()])

#lemmatazing words
data2['reviews.text'] = data2['reviews.text'].apply(lambda x : [WordNetLemmat
izer().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
data2['reviews.text'] = data2['reviews.text'].apply(lambda x: [item for item
in x if item not in stop])
```

In [42]:

```
data2["reviews.text"] = data2["reviews.text"].apply(lambda x: str(' '.join(x)
)) #joining all tokens
```

For data3

In [43]:

```
#lower case all text
data3["reviews.text"]=data3["reviews.text"].str.lower()

#tokenization of words
data3['reviews.text'] = data3.apply(lambda row: word_tokenize(row['reviews.te
xt']), axis=1)

# all the characters in the string are alphabetic only
data3["reviews.text"] = data3['reviews.text'].apply(lambda x: [item for item
in x if item.isalpha()])

#lemmatazing words
data3['reviews.text'] = data3['reviews.text'].apply(lambda x : [WordNetLemmat
izer().lemmatize(y) for y in x])

#removing useless words
stop = stopwords.words('english')
data3['reviews.text'] = data3['reviews.text'].apply(lambda x: [item for item
in x if item not in stop])
```

In [44]:

```
data3["reviews.text"] = data3["reviews.text"].apply(lambda x: str(' '.join(x)
)) #joining all tokens
```

In [45]:

```
data2.head()
data3.head()
```

```
frames = [data, data2, data3]
finalData = pd.concat(frames)
```

In [48]:

```
type(frames)
```

```
finalData.count()
```

```
finalData.head()
```

```
plt.figure(figsize=(10,5))
sns.countplot(finalData['reviews.rating'], alpha = 1)
plt.title('rating counts')
```

2 Labels

In [112]:

```
# Mapping sentiment postive, negative to the reviews.rating
sentiment = {1: 0,
        2: 0,
        3: 0,
        4: 1,
        5: 1}
finalData['sentiment'] = finalData['reviews.rating'].map(sentiment)
```

In [53]:

```
finalData.head()
```

```
finalData["sentiment"].value_counts() # 1: positive sentiment, 0: negative
sentiment
```

Out[54]:

```
building tfidf matrix to train models
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [56]:

```
vectorizer =TfidfVectorizer(max_df=0.9)
text = vectorizer.fit_transform(finalData["reviews.text"])
```

In [57]:

```
text
```

Model Selection

In [116]:

```
from sklearn.model_selection import train_test_split
```

In [117]:

```
from sklearn.metrics import classification_report
```

In [118]:

```
x_train, x_test, y_train, y_test = train_test_split(text, finalData["sentiment"],
                                                    test_size=0.3,
                                                    random_state=101)
```
In [153]:
```
# we will try logistic regression first
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=1)
classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
y_pred_tr = classifier.predict(x_train)

print('Test accuracy', sum(y_test == y_pred)/len(y_test))

print("Classification Report", '\n')
print(classification_report(y_test, y_pred))

from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier()
classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
y_pred_tr = classifier.predict(x_train)
```
In [123]:
```
print("Classification Report", '\n')
print(classification_report(y_test, y_pred))

print('Test accuracy', sum(y_test == y_pred)/len(y_test))
```

3 Labels

Trying with neutral, negative and positive

In [124]:
```
sentiment = {1: 0,
             2: 0,
             3: 1,
             4: 2,
             5: 2}
finalData['sentiment'] = finalData['reviews.rating'].map(sentiment)
```
In [126]:
```
finalData.head()

finalData['sentiment'].value_counts() # 2: positive sentiment, 1: neutral
sentiment 0: negative sentiment

# building tfidf matrix to train models
vectorizer1 =TfidfVectorizer(max_df=0.9)
text1 = vectorizer.fit_transform(finalData["reviews.text"])
```
In [134]:

```
x_train, x_test, y_train, y_test = train_test_split(text1, finalData["sentime
nt"],
                                                     test_size=0.3,
                                                     random_state=101)
```

In [149]:

```
# we will again try to run logistic regression model first
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=1)
classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
y_pred_tr = classifier.predict(x_train)
```

```
print('Test accuracy', sum(y_test == y_pred)/len(y_test))
```

```
print("Classification Report(Test)")
print(classification_report(y_test, y_pred))
```

```
# we will again run random forest
classifier = RandomForestClassifier()
classifier.fit(x_train, y_train)
y_pred = classifier.predict(x_test)
y_pred_tr = classifier.predict(x_train)
```

In [148]:

```
print('Test accuracy', sum(y_test == y_pred)/len(y_test))
```

```
print("Classification Report", '\n')
print(classification_report(y_test, y_pred))
```