Artificial Intelligence Project

Amazon reviews sentiment analysis

Submitted by

Titiksha Sahai 05901032017

Payal Mohanty 04901032017

Ashwini H Purthy 05501032017

Under the Supervision of

Mr. Rishabh Kaushal

Assistant Professor

Department of Information Technology

Department of Information Technology

Indira Gandhi Delhi Technical University for Women

Kashmere Gate, Delhi – 110006

Contents

CHAPTER 1

1. INTRODUCTION

Sentiment analysis is a text analysis method that detects polarity (e.g. a *positive* or *negative* opinion) within text, whether a whole document, paragraph, sentence, or clause.

Understanding people's emotions is essential for businesses since customers are able to express their thoughts and feelings more openly than ever before. By automatically analyzing customer feedback, from survey responses to social media conversations, brands are able to listen attentively to their customers, and tailor products and services to meet their needs.

1.1PROBLEM STATEMENT:

Sentiment analysis of product reviews, an application problem, has recently become very popular in text mining and computational linguistics research. Here, we want to study the correlation between the Amazon product reviews and the rating of the products given by the customers. We use traditional machine learning algorithms including Naive Bayes analysis and Logestic Regression. By comparing these results, we could get a better understanding of these algorithms. They could also act as a supplement to other fraud scoring detection methods.

CHAPTER 2:

2. LITERATURE SURVEY:

2.1 RESEARCH QUESTIONS:

- What are the most reviewed Amazon products
- What are the initial and current number of customer reviews for each product?
- How do the reviews in the first 90 days after a product launch compare to the price of the product?
- How do the reviews in the first 90 days after a product launch compared to the days available for sale?

CHAPTER 3

3. PROPOSED METHODOLOGY:

3.1 DATASET:

Number of instances - 34660

Number of attributes - 21

Label information - Label present

ATTRIBUTE NAME	ATTRIBUTE TYPE
Id	Non null object
Name	Non null object
Asins	Non null object
Brand	Non null object
Categories	Non null object
Keys	Non null object
Manufacturer	Non null object
reviews.date	Non null object
reviews.dateAdded	Non null object
reviews.dateSeen	Non null object
reviews.didPurchase	Non null object
reviews.doRecommend	Non null object
reviews.id	Non null float64

reviews.rating	Non null float64
reviews.sourceURLs	Non null object
reviews.text	Non null object
reviews.title	Non null object
reviews.userCity	Non null float64
reviews.userProvince	Non null float64
reviews.username	Non null object

3.2 ATTRIBUTE DESCRIPTION:

- **Id** : unique non null object associated with each entry of dataset
- **Asins**: Amazon Standard Identification Number is a 10-charcter alphanumeric unique identifier that's assigned by Amazon.com and its partners. It's primarily used for product-identification within their product catalog.
- **Brand** : name of the brand which manufactured the product
- **Categories** : the category in which the product belongs
- **Keys**: uniquely identifies the record
- **Manufacturer**: name of the manufacturer of the product
- **Reviews.date** : contains information regarding the review timing
- **Reviews.dateAdded** : date at which the review was added
- **Reviews.dateSeen** : date at which the review was seen
- **Reviews.didPurchase** : verifies if the reviews is added by a customer
- **Reviews.doRecommend** : does the customer recommend the product(true/false)
- **Reviews.Id** : unique object associated with each review
- **Reviews.numHelpful** : shows the helpfulness of a review
- **Reviews.rating** : Rating of a product associated with each review (1-5)
- **Reviews.sourceURL** : URL source of each review in the dataset

- **Reviews.text** : text in each product review by customer
- **Reviews.title** : title of each review by customer
- **Reviews.userCity** : City name of the customer who writes the review
- **Reviews.userProvince** : Province name of the customer who writes the review
- **Reviews.username** : Username of the customer who add the review

3.3 DETAILS ABOUT THE ORGANIZATION:

The dataset is collected from kaggle.com. Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

Kaggle got its start in 2010 by offering machine learning competitions and now also offers a public data platform, a cloud-based workbench for data science, and Artificial Intelligence education. Its key personnel were Anthony Goldbloom and Jeremy Howard. Nicholas Gruen was founding chair succeeded by Max Levchin. Equity was raised in 2011 valuing the company at $25 million. On 8 March 2017, Google announced that they were acquiring Kaggle.

3.4 DATA PREPROCESSING:

3.4.1 DATA FILES COMBINING:

Data is available in the form of 1429_1.csv containing 34,000 consumer reviews for Amazon products like the Kindle, Fire TV Stick, and more provided by Datafiniti's Product Database. The dataset includes basic product information, rating, review text, and more for each product.

3.4.2 DATA CLEANING:

- Drop reviews.userCity, reviews.userProvince, reviews.id, and reviews.didPurchase since these values are floats (for exploratory analysis only)
- Not every category have maximum number of values in comparison to total number of values
- reviews.text category has minimum missing data (34659/34660) -> Good news!

- We need to clean up the name column by referencing asins (unique products) since we have 7000 missing values.

Chapter 4

4. DATA EXPLORATION

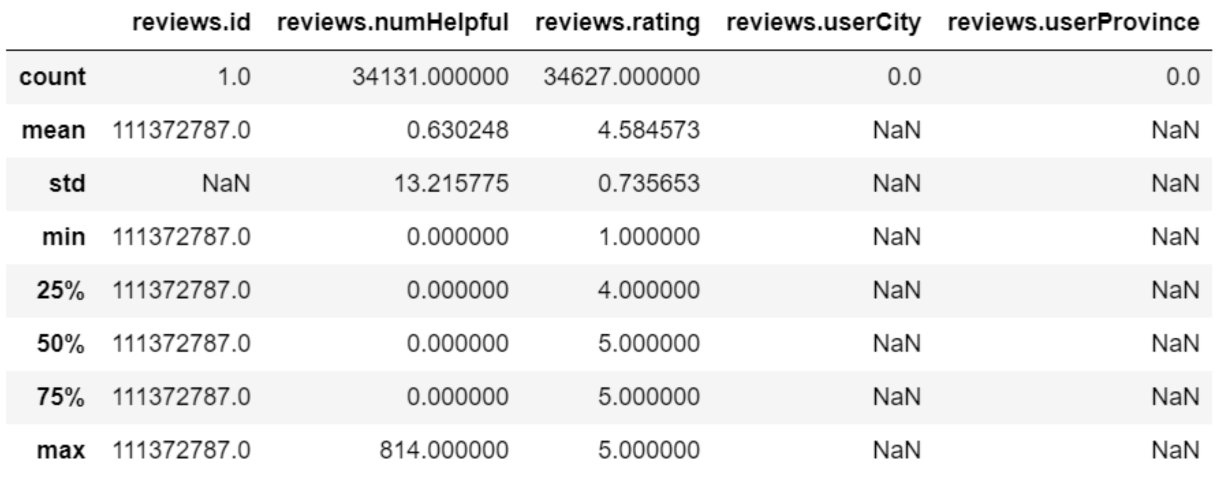
4.1 METRICS

Based on the descriptive statistics above, we see the following:

- Average review score of 4.58, with low standard deviation
  - Most review are positive from 2nd quartile onwards

- The average for number of reviews helpful (reviews.numHelpful) is 0.6 but high standard deviation
  - The data are pretty spread out around the mean, and since can't have negative people finding something helpful, then this is only on the right tail side
  - The range of most reviews will be between 0-13 people finding helpful (reviews.numHelpful)

- The most helpful review was helpful to 814 people
  - This could be a detailed, rich review that will be worth looking at.



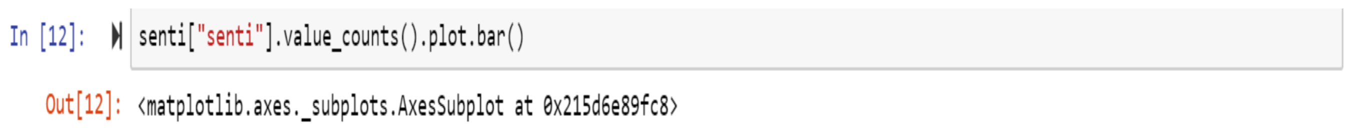
	reviews.id	reviews.numHelpful	reviews.rating	reviews.userCity	reviews.userProvince
count	1.0	34131.000000	34627.000000	0.0	0.0
mean	111372787.0	0.630248	4.584573	NaN	NaN
std	NaN	13.215775	0.735653	NaN	NaN
min	111372787.0	0.000000	1.000000	NaN	NaN
25%	111372787.0	0.000000	4.000000	NaN	NaN
50%	111372787.0	0.000000	5.000000	NaN	NaN
75%	111372787.0	0.000000	5.000000	NaN	NaN
max	111372787.0	814.000000	5.000000	NaN	NaN

4.2 VISUALIZATIONS

4.2.1 COUNT OF REVIEWS

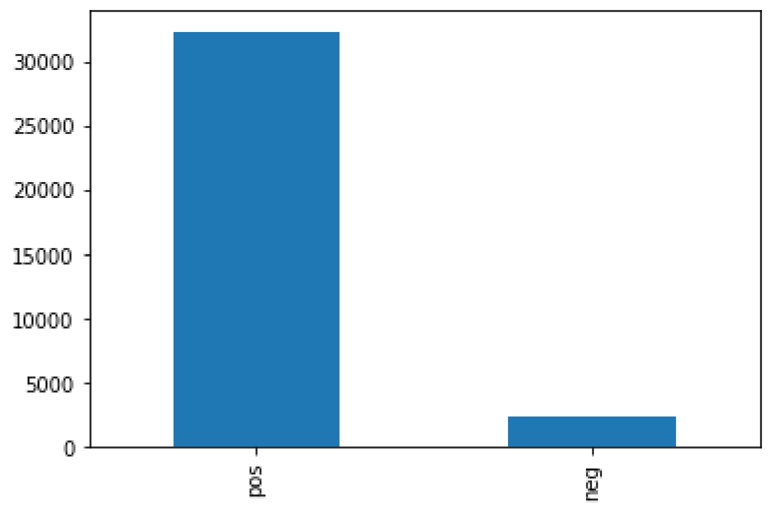
This helps us understand the count of positive and negative reviews present in the dataset.

CODE:



```
In [12]:  senti["senti"].value_counts().plot.bar()

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x215d6e89fc8>
```

OUTPUT:

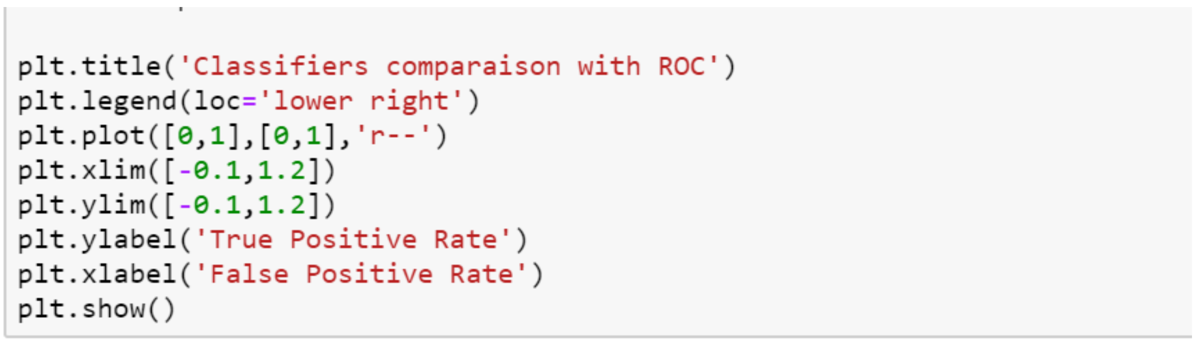


As we can see data is unbalanced so this will create problem for model but, this data as it is and will predict our reviews.

4.2.2 CLASSIFIER COMPARISON WITH ROC

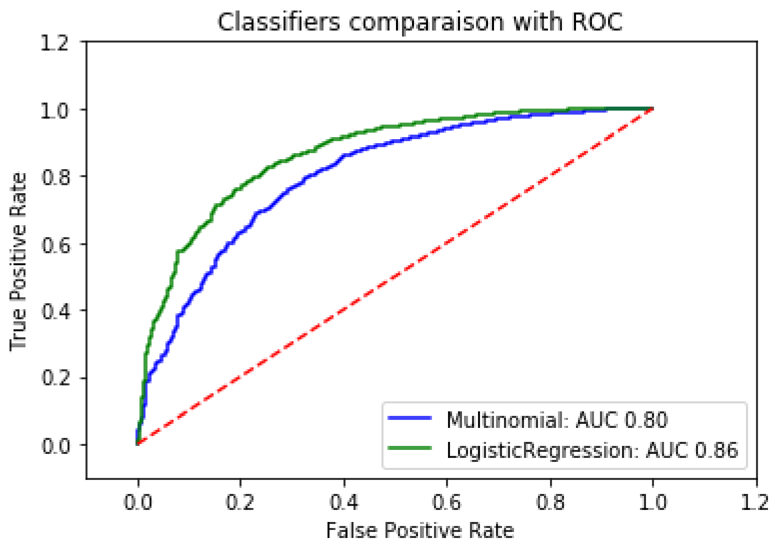
ROC curves give us the ability to assess the performance of the classifier over its entire operating range. The Area under the Curve (AUC) of a perfect classifier is 1.0. Here we found the AUC for multinomial as 0.80 and for logestic regression as 0.86. Hence, logestic regression is found to be a better classification algorithm as compared to multinomial naïve bayes.

CODE:



```
plt.title('Classifiers comparaison with ROC')
plt.legend(loc='lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([-0.1,1.2])
plt.ylim([-0.1,1.2])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

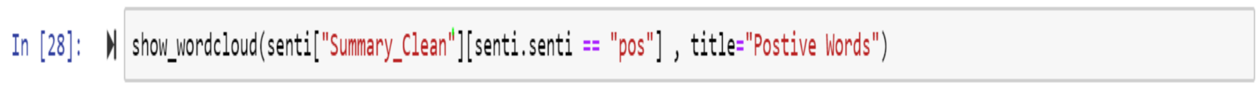
OUTPUT:



4.2.3 POSITIVE WORDS :

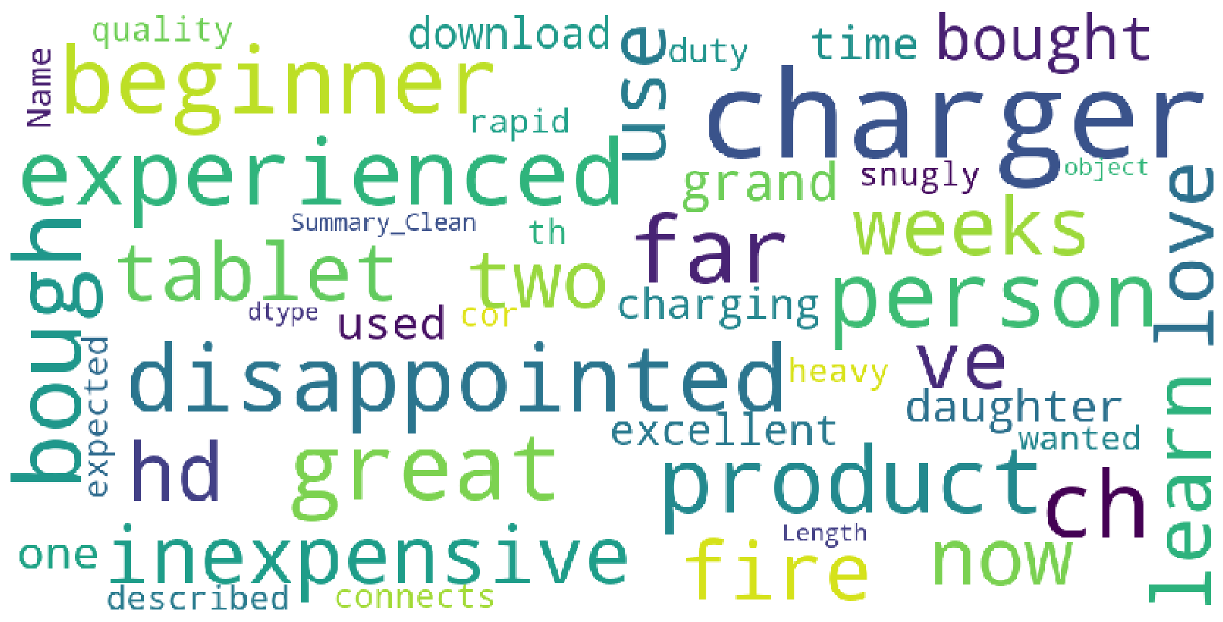
This helps us to get a wordcloud of al the positive words used in the dataset. This allows us to assess which positive words are more prominent and also get an idea of the tone with which they are used.
For instance, Green means that the words are mainly used in positive sentiment.

CODE:



```
In [28]:  show_wordcloud(senti["Summary_Clean"][senti.senti == "pos"] , title="Postive Words")
```

OUTPUT:

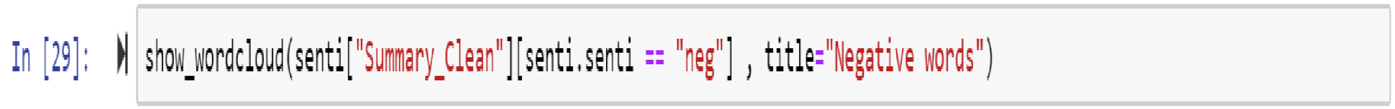


Postive Words

4.2.4 NEGATIVE WORDS

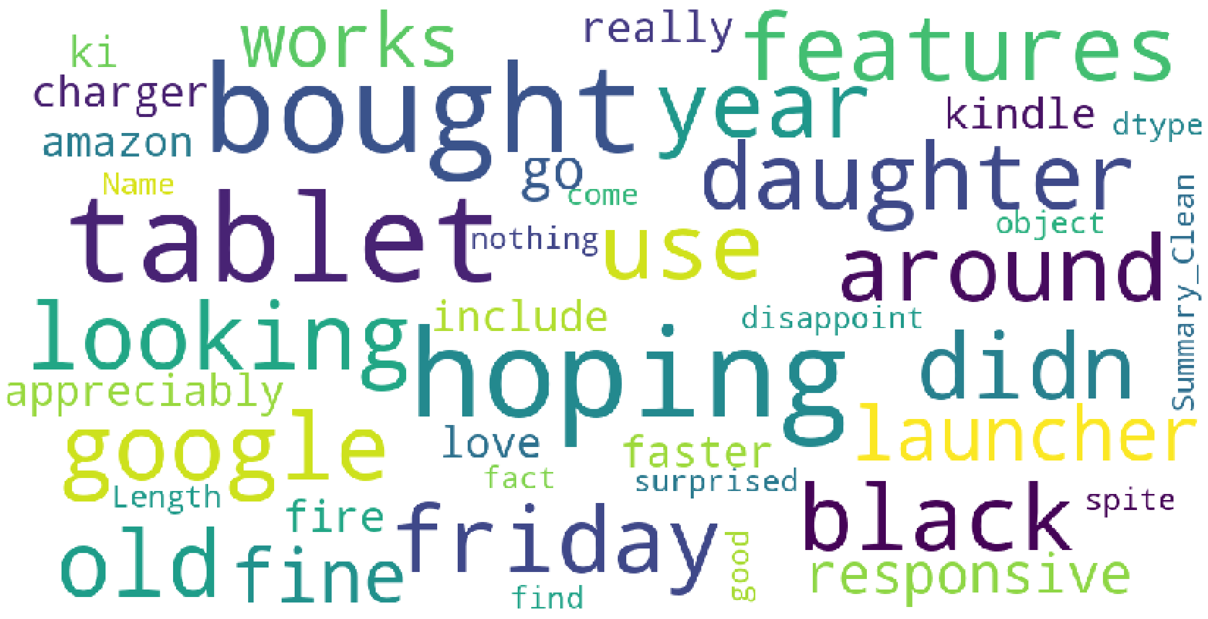
This helps us to get a wordcloud of all the negative words in the dataset. . This allows us to assess which negative words are more prominent and also get an idea of the tone with which they are used.
For instance, Red means that the words are mainly used in negative sentiment.

CODE:



```
In [29]:  show_wordcloud(senti["Summary_Clean"][senti.senti == "neg"] , title="Negative words")
```

OUTPUT:



Negative words

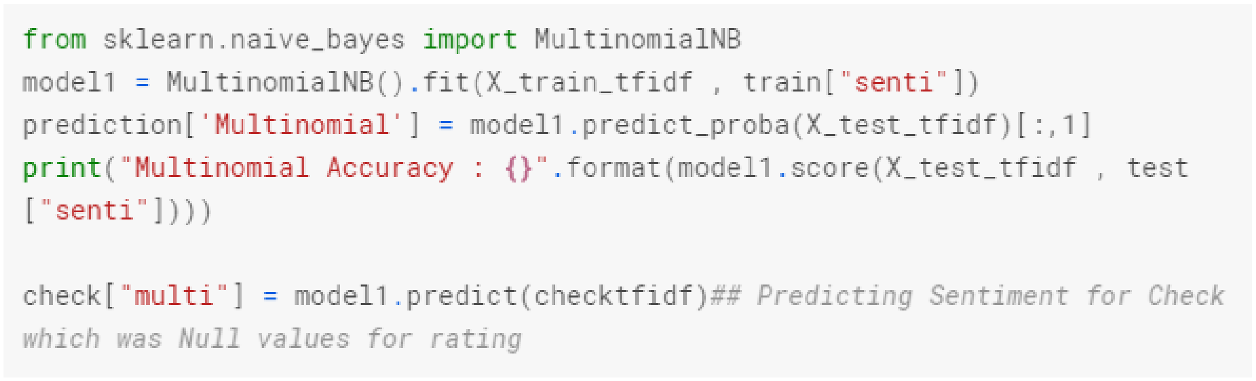
CHAPTER 5

5.1ALGORITHMS

5.1.1 MULTINOMINAL NAIVE BAYES

- Multinomial Naive Bayes is most suitable for word counts where data are typically represented as word vector counts (number of times outcome number X[i,j] is observed over the n trials), while also ignoring non-occurrences of a feature i

- Naive Bayes is a simplified version of Bayes Theorem, where all features are assumed conditioned independent to each other (the classifiers), P (x|y) where x is the feature and y is the classifier.



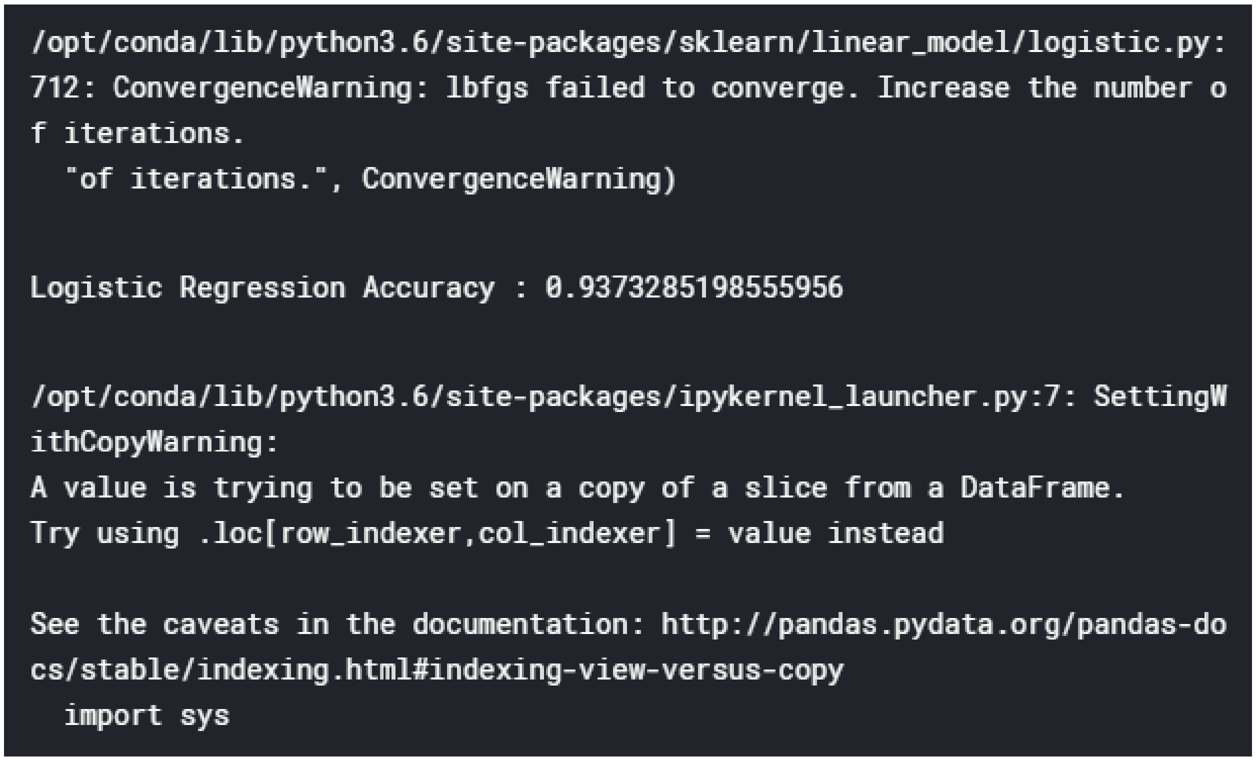
```
from sklearn.naive_bayes import MultinomialNB
model1 = MultinomialNB().fit(X_train_tfidf , train["senti"])
prediction['Multinomial'] = model1.predict_proba(X_test_tfidf)[:,1]
print("Multinomial Accuracy : {}".format(model1.score(X_test_tfidf , test
["senti"])))

check["multi"] = model1.predict(checktfidf)## Predicting Sentiment for Check
which was Null values for rating
```

```
Multinomial Accuracy : 0.9329963898916968


/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:6: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-do
cs/stable/indexing.html#indexing-view-versus-copy
```

5.1.2 LOGISTIC REGRESSION CLASSIFIER

It is used to determine the output or result when there are one or more than one independent variables. The output value can be in form of 0 or 1 i.e. in binary form.

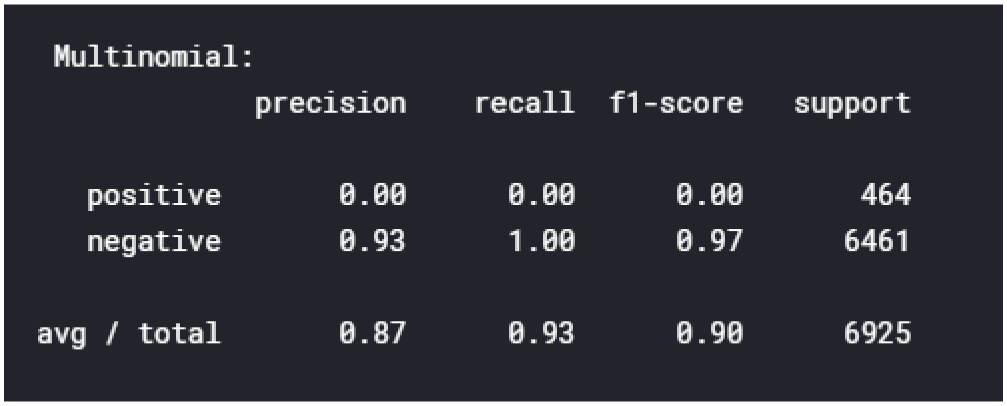
```
from sklearn import linear_model
logreg = linear_model.LogisticRegression(solver='lbfgs' , C=1000)
logistic = logreg.fit(X_train_tfidf, train["senti"])
prediction['LogisticRegression'] = logreg.predict_proba(X_test_tfidf)[:,1]
print("Logistic Regression Accuracy : {}".format(logreg.score(X_test_tfidf
, test["senti"])))

check["log"] = logreg.predict(checktfidf)## Predicting Sentiment for Check w
hich was Null values for rating
```



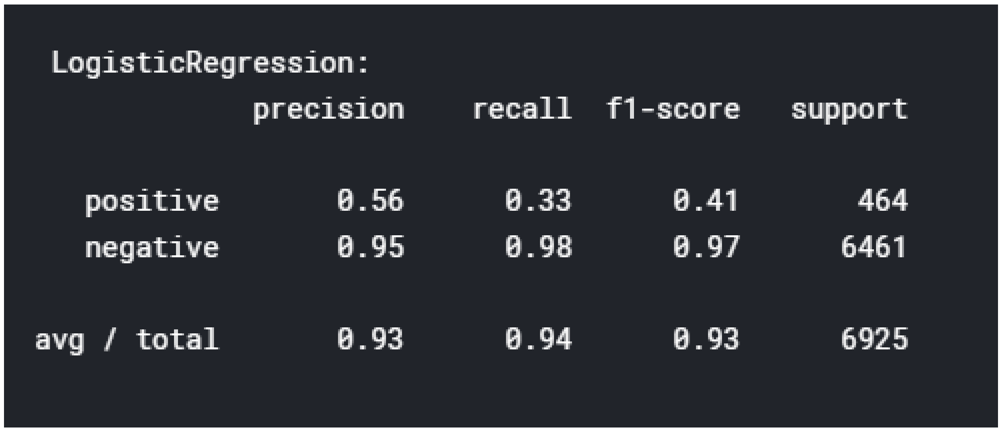
```
/opt/conda/lib/python3.6/site-packages/sklearn/linear_model/logistic.py:
712: ConvergenceWarning: lbfgs failed to converge. Increase the number o
f iterations.
  "of iterations.", ConvergenceWarning)


Logistic Regression Accuracy : 0.9373285198555956


/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:7: SettingW
ithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-do
cs/stable/indexing.html#indexing-view-versus-copy
  import sys
```

5.2 CONCLUSION:

We used Multinomial Naive Bayes and Logestic Regression as our Classifier which are most suitable for word counts where data are typically represented as word vector counts. Then we tested other models to find out that both the models performed very well (>90%). Although Logestic Regression has an edge over multinomial naïve bayes with better accuracy and precision.



```
Multinomial:
             precision    recall  f1-score   support

   positive       0.00      0.00      0.00       464
   negative       0.93      1.00      0.97      6461

avg / total       0.87      0.93      0.90      6925
```



```
LogisticRegression:
             precision    recall  f1-score   support

   positive       0.56      0.33      0.41       464
   negative       0.95      0.98      0.97      6461

avg / total       0.93      0.94      0.93      6925
```

Also from the analysis above in the classification report, we can see that products with lower reviews are not significant enough to predict these lower rated products are inferior. On the other hand, products that are highly rated are considered superior products, which also performs well and should continue to sell at a high level.

As a result, we need to input more data in order to consider the significance of lower rated product, in order to determine which products should be dropped from Amazon's product roster.

The good news is that despite the skewed dataset, we were still able to build a robust Sentiment Analysis machine learning system to determine if the reviews are positive or negative. This is possible as the machine learning system was able to learn from all the positive, neutral and negative reviews, and fine tune the algorithm in order to avoid bias sentiments.

In conclusion, although we need more data to balance out the lower rated products to consider their significance, however we were still able to successfully associate positive, neutral and negative sentiments for each product in Amazon's Catalog.

CHAPTER 6

6.1 FUTURE WORK

From future perspective, we would like to extend this project by implementing some machine learning algorithms for applications like election results, product ratings, movies' outcomes and running the project on clusters to expand its functionalities. Moreover, we would like to make a web application for users to input keywords and get analyzed results. In this project, we have worked only with unigram models, but we would like to extend it to bigram and further which will increase linkage between the data and provide accurate sentiment analysis results. Computation of overall tweet score can be done for a single keyword which can provide an overall sentiment of public regarding a topic.

6.2 BIBLIOGRAPHY

- *Consumer reviews of amazon product* : data source (1429_1.csv) from kaggle.com
- Mick Zhang; *Amazon Reviews using Sentiment analysis*
- Wanliang Tan, Xinyu Wang, Xinyu Xu ; *Sentiment Analysis for Amazon Reviews*
- Stuart Russell, Peter Norvig; *Artificial Intelligence – A modern Approach (Third Edition)*