VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

NATURAL LANGUAGE PROCESSING

SWE1017

SENTIMENT ANALYSIS AND EDA ON AMAZON CUSTOMER REVIEWS REPORT

DONE BY

SHIVANI GOKULRAM NAIDU 19MIA1006

MADASU DEEPIKA 19MIA1066

HARINISRI G. 19MIA1069

ABSTRACT:

Amazon is an American multinational corporation that focuses on ecommerce, cloud computing, digital streaming, and artificial intelligence products. But it is mainly known for its e-commerce platform which is one of the biggest online shopping platforms today. So having such a large customer base, it would be great to do sentiment analysis on their reviews.

Exploratory Data Analysis is the process of exploring data, generating insights,testing hypotheses, checking assumptions and revealing underlying hidden patterns in the data.

Sentiment analysis is a powerful marketing tool that enables product managers to understand customer emotions in their marketing campaigns. It is an important factor when it comes to product and brand recognition, customer loyalty, customer satisfaction, advertising and promotion's success, and product acceptance.

We plan on performing a comparative study between different machine learning models to perform sentiment analysis on the customer reviews of Amazon products. We will be performing Long short term memory (LSTM) in order to find the accuracy of the model and it has achieved to be 97 %.

INTODUCTION:

Recent years have seen an increasing amount of research efforts expanded in understanding sentiment in textual resources

One of the subtopics of this research is called sentiment analysis or opinion mining, which is, given a bunch of text, we can computationally study peoples opinions, appraisals, attitudes, and

emotions toward entities, individuals, issues, events, topics and their attributes. Applications of this technique are diverse.

For example, businesses always want to find public or consumer opinions and emotions about their products and services. Potential customers also want to know the opinions and emotions of existing users before they use a service or purchase a product. Last but not least, researchers uses these information to do an in-depth analysis of market trends and consumer opinions, which could potentially lead to a better prediction of the stock market.

However, saying this, to find and monitor opinion sites on the Web and distill the information contained in them remains a formidable task because of the proliferation of diverse sites. Each site typically contains a huge volume of opinionated text that is not always easily deciphered in long forum postings and blogs. The average human reader will have difficulty identifying relevant sites and accurately summarizing the information and opinions contained in them. Besides, to instruct a computer to recognize sarcasm is indeed a complex and challenging task given that at the moment, computer still cannot think like human beings.

The objective of this paper is to classify the positive and negative reviews of the customers over different products and build a supervised learning model to polarize large amounts of reviews. Our dataset consists of customers' reviews and ratings, which we got from Consumer Reviews of Amazon products. We extracted the features of our dataset and built LSTM model based on that. We take the accuracy of this model and got a better understanding of the polarized attitudes towards the products.

DATASET DESCRIPTION:

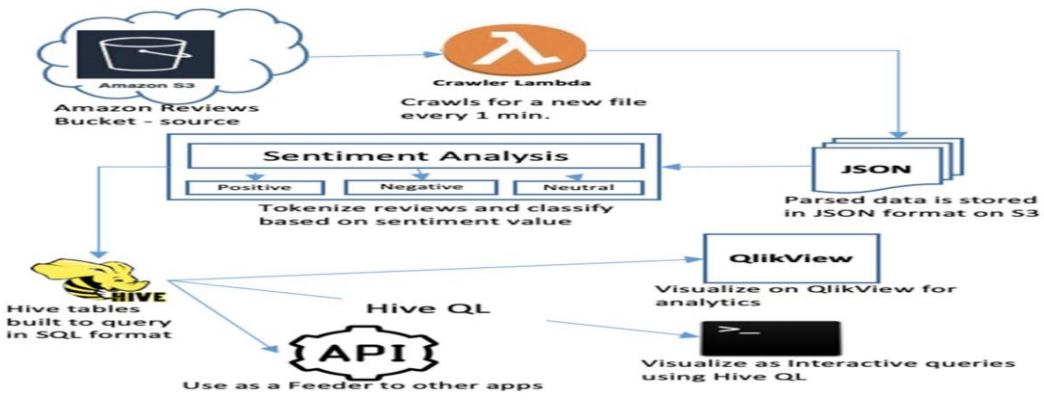
The Dataset is downloaded from Kaggle.

It contains 34,000 consumer reviews for Amazon products like the Kindle, Fire TV Stick, and more. It includes basic product information, rating, review text, and more for each product.

This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014.This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs). This is intended to facilitate study into the properties (and the evolution) of customer reviews potentially including how people evaluate and express their experiences with respect to products at scale. (130M+ customer reviews)

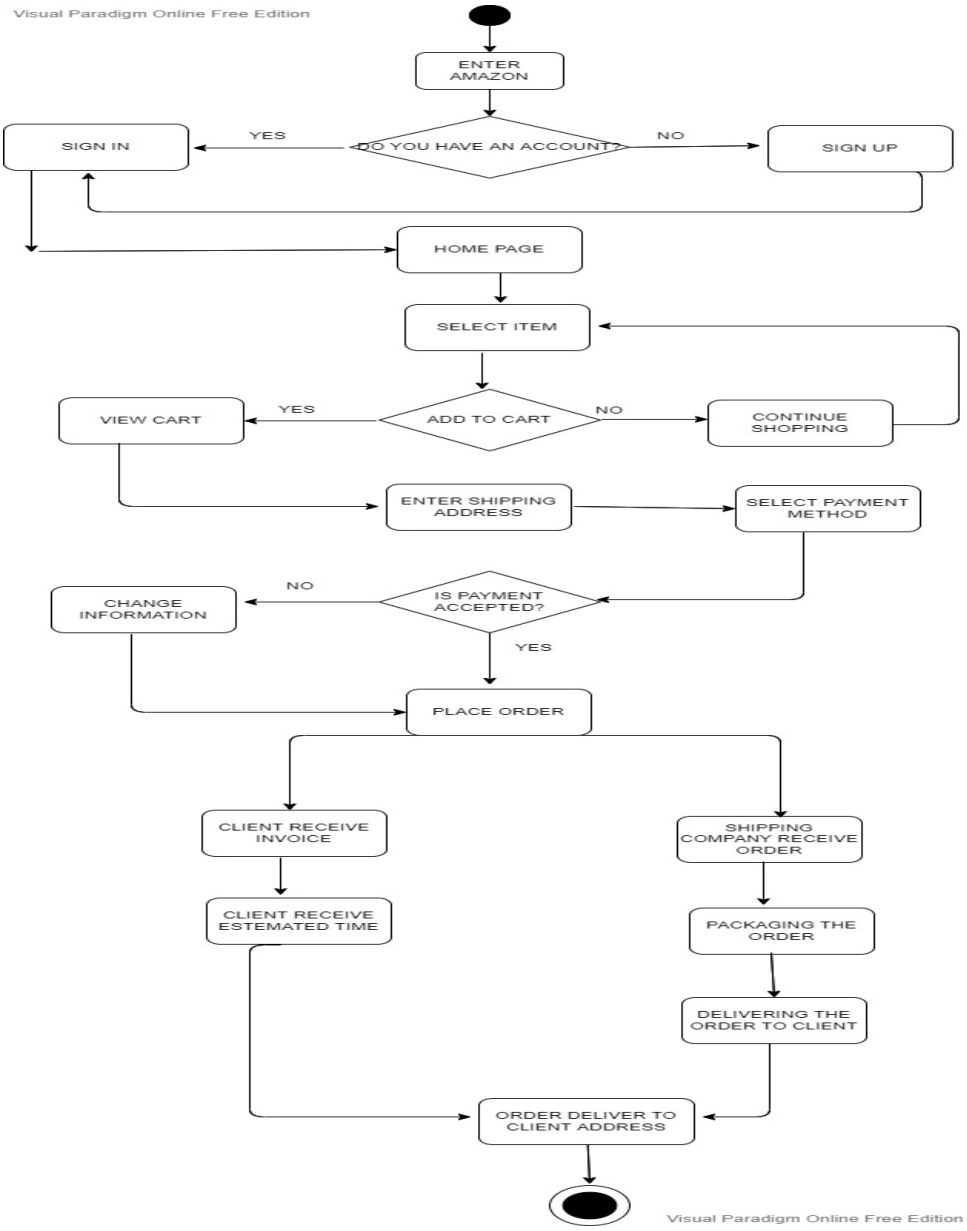
A collection of reviews about products in multiple languages from different Amazon marketplaces, intended to facilitate analysis of customers' perception of the same products and wider consumer preferences across languages and countries. (200K+ customer reviews in 5 countries).

ARCHITECTURE:

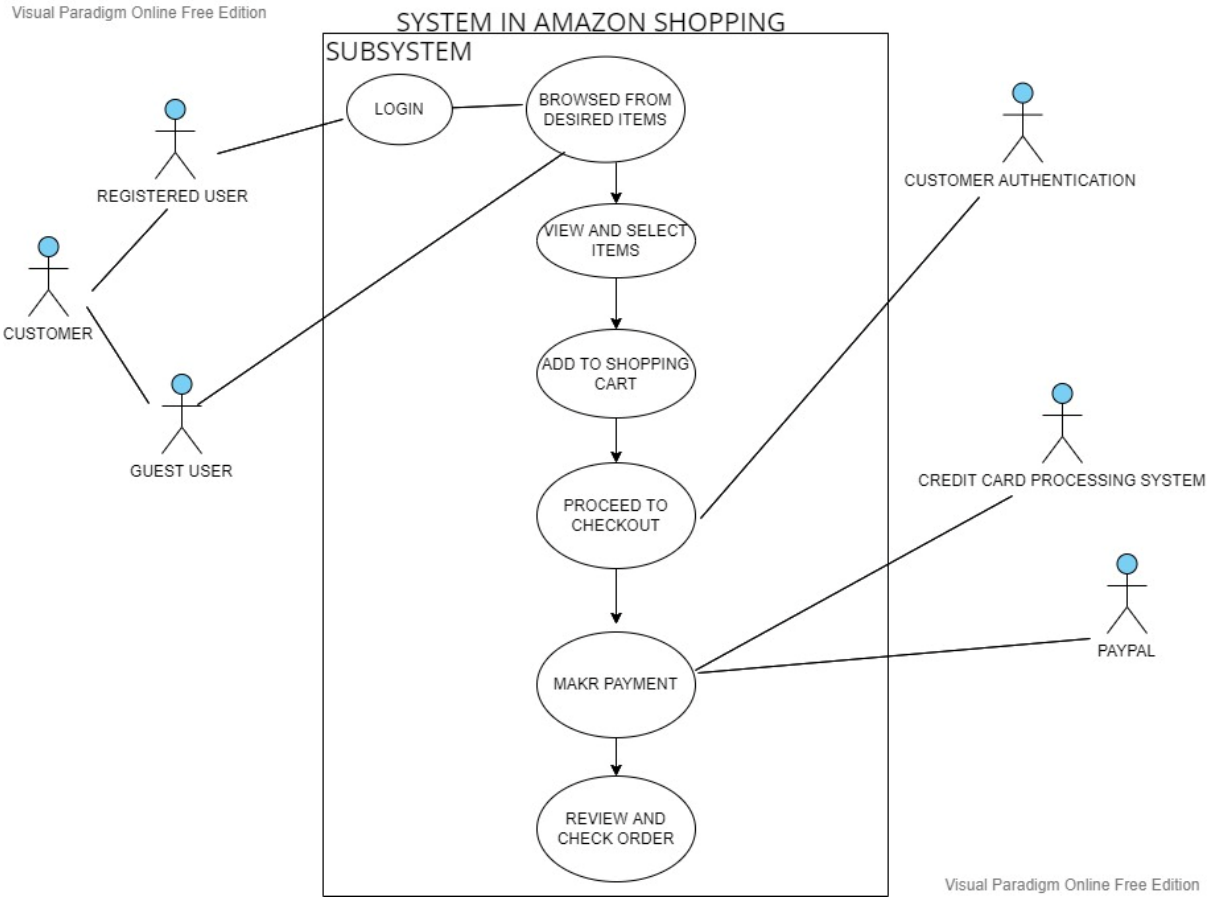


BLOCK DIAGRAMS: We have created the necessary amazon block diagrams using visual paradigm software.

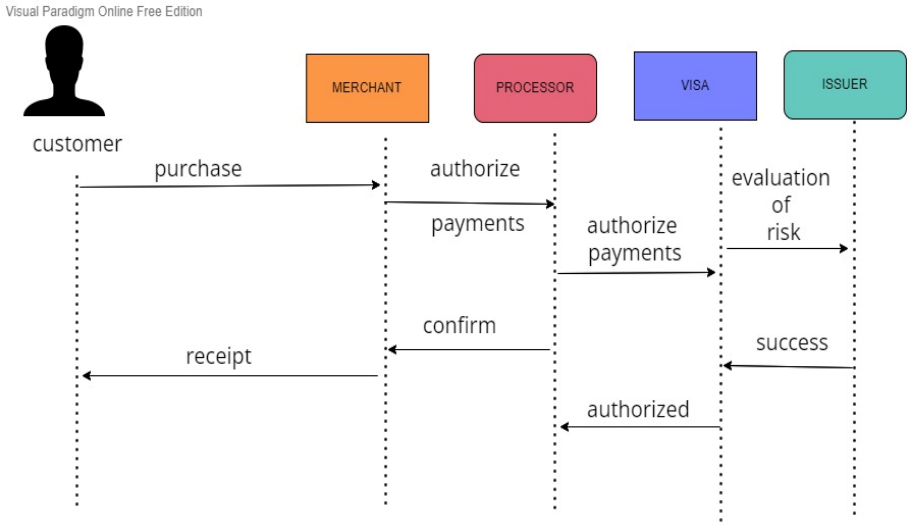
(I) ACTIVITY DIAGRAM:



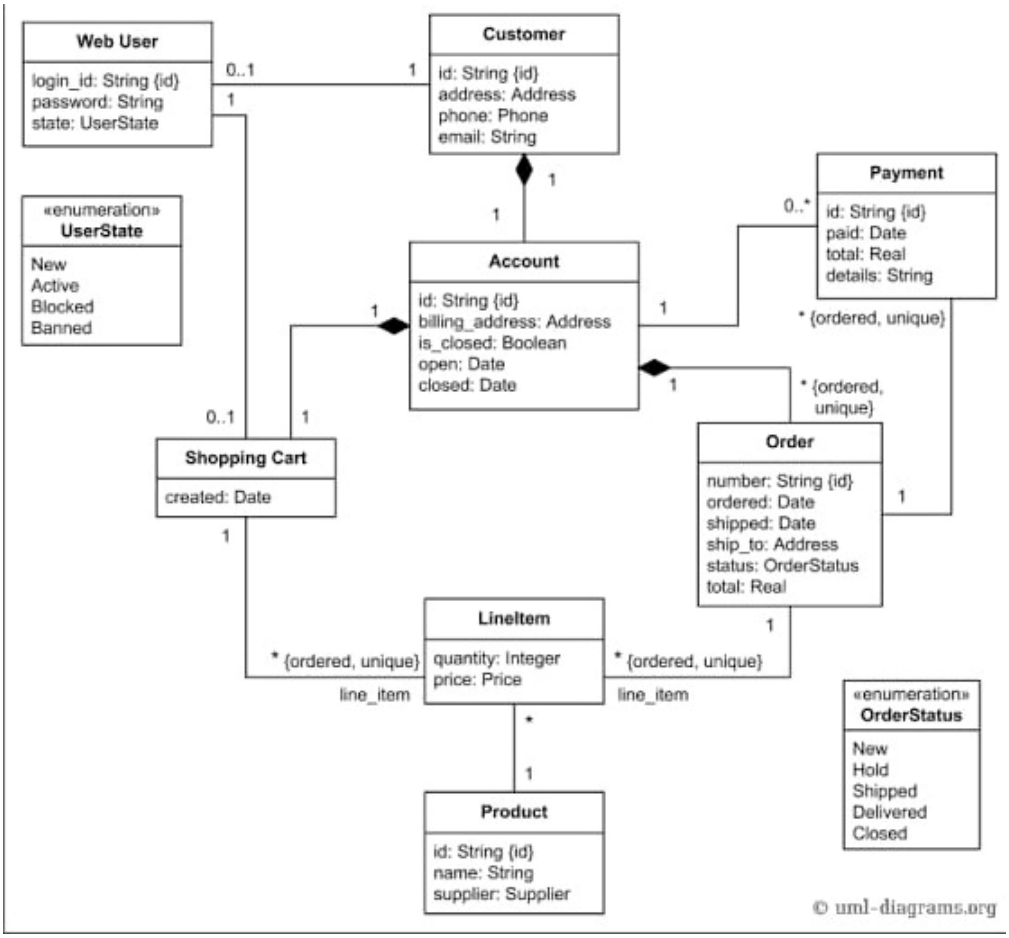
(II) USER CASE DIAGRAM:



(III) SEQUENCE DIAGRAM:



(IV) CLASS DIAGRAM:



METHODOLGY:

DATA COLLECTION

This is a list of over 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.

The full dataset is available through Datafiniti/ kaggle.

DATA ANALYSIS

PREPROCESSING

The dataset contains 34,660 rows and 21 columns. But we only need information such as product name, review text, user recommendation (binary), and the number of people that found a

review helpful. Therefore, we are dropping other columns and reducing the dataset to only four columns, i.e., 'name', 'reviews.text','reviews.doRecommend', and 'reviews.numHelpful'. There are a few null values in the dataset. So, we drop these null values.We are only considering those products that have at least 500 reviews. This is doneto make sure that we have a sufficient number of reviews for each product. Here, we will use lambda functions with filter() to filter our dataset.

Now, we are left with eight products. Also, the 'reviews.doRecommend' column contains values in the form of True-False and' reviews.numHelpful' contains floatingpoint numbers, which is not possible. Therefore, we are converting these columns into integers

CLEANING THE TEXT DATA

Generally, text data contains a lot of noise either in the form of symbols or in the form of punctuations and stop words. Therefore, it becomes necessary to clean the text, not just for making it more understandable but also for getting better insights.

We have four columns in our dataset out of which two columns ('name','reviews.text') contain textual data. we have some contractions like "It's", numberslike "3" and punctuations like ",", "!" and "." present in the reviews. We handle these by performing the below operations:

- **Expand contractions**
- **Lowercase the reviews**
- **Remove digits and words containing digits**
- **Remove punctuations**
- **Stopwords Removal**
- **Lemmatization**

• Create Document Term Matrix.

Sentiment analysis is a powerful marketing tool that enables product managers to understand customer emotions in their marketing campaigns. It is an important factor when it comes to product and brand recognition, customer loyalty, customer satisfaction, advertising and promotion's success, and product acceptance.

It quantifies the emotional intensity of words and phrases within a text. Sentiment analysis tools will process a unit of text and output quantitative scores to indicate +ve/-ve. NLTK VADER sentiment analysis tool generates +ve, -ve and neutral sentiment scores for a given input. Sentiment analysis is essential for businesses to gauge customer response.

A sentiment analysis API uses natural language processing (NLP) tasks to not only identify aspects of the products from the Amazon reviews but also enable brands to look beyond star ratings. Amazon review data analysis can give insightful customer information that can be harnessed for product betterment. The reviews are unstructured. In other words, the text is unorganized. Sentiment analysis, however, helps us make sense of all this unstructured text by automatically tagging it. Sentiment analysis helps us to process huge amounts of data in an efficient and costeffective way.

Methods used in sentimental analysis: tokenization and Long short term memory.

Tokenization is the first step in text analytics. The process of breaking down a text paragraph into smaller chunks such as words or sentence is called Tokenization. Token is a single entity that is building blocks for sentence or paragraph. Tokenization is the process of breaking text into pieces, called tokens, and

ignoring characters like punctuation marks („ “ ‘) and spaces. spaCy‘s tokenizer takes input in form of unicode text and outputs a sequence of token objects.Import spaCy and its English-language model. Assign the review text string to text. Using nlp(text), will process that text in spaCy and assign the result to a variable called doc. spaCy stores tokenized text as a doc.

A sequence dataset such as Amazon review dataset are best fit with LSTM as state transducers are involved in each of the feedback state of the model. As the dataset under consideration is unsegmented, LSTM has its upper hold when compared to other deep learning algorithms. To work with the LSTM, one cells output is fed as input for the other cell. This sequence is fabricated further for this proposed model in Keras for the purpose of classification.

RESULTS AND DISCUSSIONS:

STEP 1: IMPORTING ALL THE REQUIRED LIBRARIES AND DATASET.

Out[4]:

	id	name	asins	brand	categories	keys	manufacturer	reviews.date	reviews.dateAdded	reviews.dateSeen	...	re
0	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon	2017-01-13T00:00:00.000Z	2017-07-03T23:33:15Z	2017-06-07T09:04:00.000Z,2017-04-30T00:45:00.000Z	...	
1	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon	2017-01-13T00:00:00.000Z	2017-07-03T23:33:15Z	2017-06-07T09:04:00.000Z,2017-04-30T00:45:00.000Z	...	
2	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon	2017-01-13T00:00:00.000Z	2017-07-03T23:33:15Z	2017-06-07T09:04:00.000Z,2017-04-30T00:45:00.000Z	...	
3	AVqkIhwDv8e3D1O-lebb	All-New Fire HD 8 Tablet, 8 HD Display, Wi-Fi,...	B01AHB9CN2	Amazon	Electronics,iPad & Tablets,All Tablets,Fire Ta...	841667104676,amazon/53004484,amazon/b01ahb9cn2...	Amazon	2017-01-13T00:00:00.000Z	2017-07-03T23:33:15Z	2017-06-07T09:04:00.000Z,2017-04-30T00:45:00.000Z	...	
		All-										

STEP 2: DATA ANLYSIS AND PREPROCESSING WHERE WE WILL BE DROPPING OUT UNNECESSARY COLUMNS.

In [5]:
```
data.drop('id',axis=1,inplace=True)
data.drop('asins',axis=1,inplace=True)
data.drop('keys',axis=1,inplace=True)
```

In [6]:
```
data.drop('manufacturer',axis=1,inplace=True)
data.drop('reviews.id',axis=1,inplace=True)
data.drop('reviews.sourceURLs',axis=1,inplace=True)
data.drop('reviews.userCity',axis=1,inplace=True)
data.drop('reviews.userProvince',axis=1,inplace=True)
data.drop('reviews.didPurchase',axis=1,inplace=True)

data.drop('reviews.dateAdded',axis=1,inplace=True)
data.drop('reviews.dateSeen',axis=1,inplace=True)
```

In [7]:
```
#remove empty rows
data = data[~data['reviews.text'].isnull()]
data = data[~data['reviews.doRecommend'].isnull()]
data = data[~data['reviews.numHelpful'].isnull()]
data = data[~data['reviews.rating'].isnull()]
```

In [8]:
```
#preprocessing
data['reviews.title'] = data['reviews.title'].apply(lambda x: str(x).lower())
data['reviews.title'] = data['reviews.title'].apply(lambda x: re.sub('\w*\d\w*','', x))
#data['reviews.title'] = data['reviews.title'].apply(lambda x: re.sub('[%s]' % re.escape(string.punctuation), '', x))

data['reviews.text'] = data['reviews.text'].apply(lambda x: str(x).lower())
data['reviews.text'] = data['reviews.text'].apply(lambda x: re.sub('\w*\d\w*','', x))
#data['reviews.text'] = data['reviews.text'].apply(lambda x: re.sub('[%s]' % re.escape(string.punctuation), '', x))
```

STEP 3: PERFORMING EDA BY FINDING SHAPE, AND INFORMATION (after preprocessing we can find that 21 columns has been reduced to 10 columns)

In [9]:
```
data.shape #shape
```

Out[9]:
```
(34064, 10)
```

In [10]:
```
data.info() #information
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 34064 entries, 0 to 34624
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   name                  27409 non-null  object 
 1   brand                 34064 non-null  object 
 2   categories            34064 non-null  object 
 3   reviews.date          34064 non-null  object 
 4   reviews.doRecommend   34064 non-null  object 
 5   reviews.numHelpful    34064 non-null  float64
 6   reviews.rating        34064 non-null  float64
 7   reviews.text          34064 non-null  object 
 8   reviews.title         34064 non-null  object 
 9   reviews.username      34062 non-null  object 
dtypes: float64(2), object(8)
memory usage: 2.9+ MB
```

STEP 4: CHECKING IF THERE IS ANY NULL VALUES AND PRODUCING HEAT MAP FOR IT

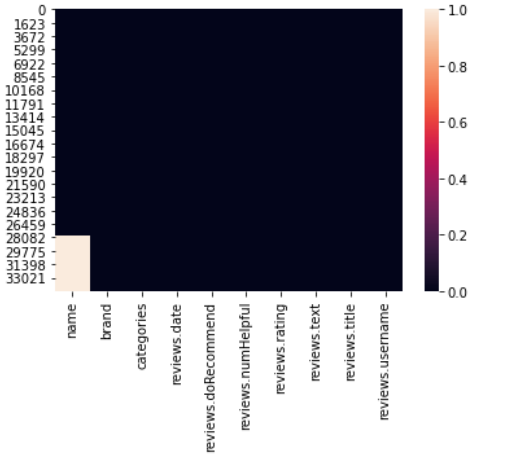
In [11]:
```
# Finding the number of unique values present in each column
data.nunique()
```

Out[11]:
```
name                    44
brand                    4
categories              23
reviews.date           943
reviews.doRecommend      2
reviews.numHelpful      57
reviews.rating           5
reviews.text         34061
reviews.title        17567
reviews.username     26312
dtype: int64
```

In [12]:
```
# Checking if any NaN is present in column or not
data.isna().any()
```

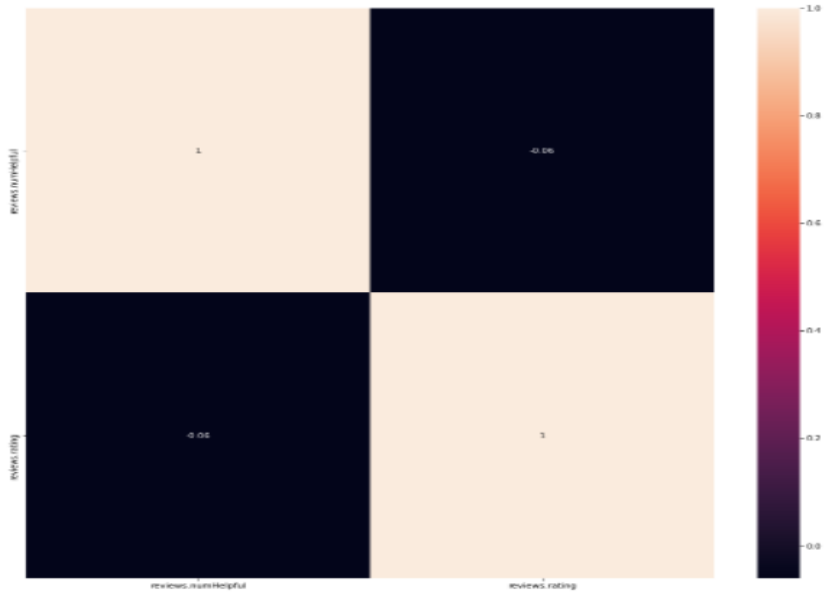
Out[12]:
```
name                  True
brand                False
categories           False
reviews.date         False
reviews.doRecommend  False
reviews.numHelpful   False
reviews.rating       False
reviews.text         False
reviews.title        False
reviews.username      True
dtype: bool
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x274b46863c8>

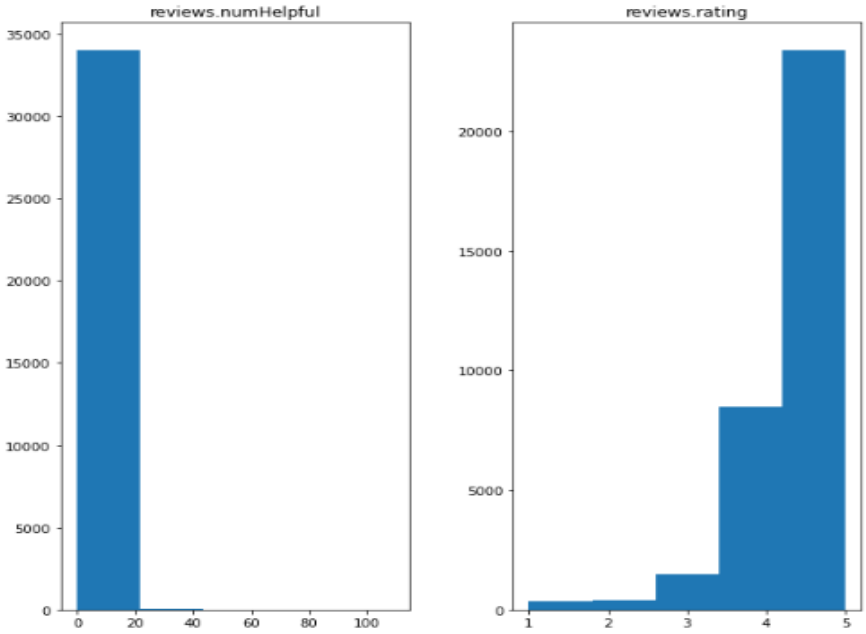


STEP 5: VISUALISING THE DATA FOR ENHANCING UNDERSTANDING.

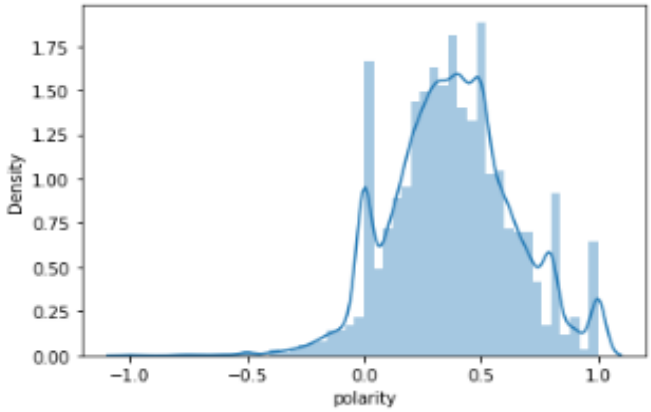
Finding the correlation between feature columns



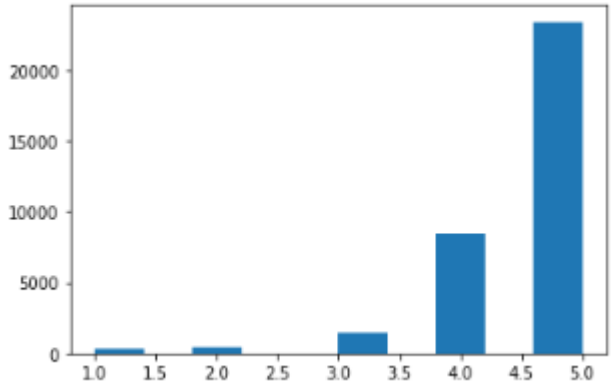
Bar graph between reviews.numhelpful and reviews.rating



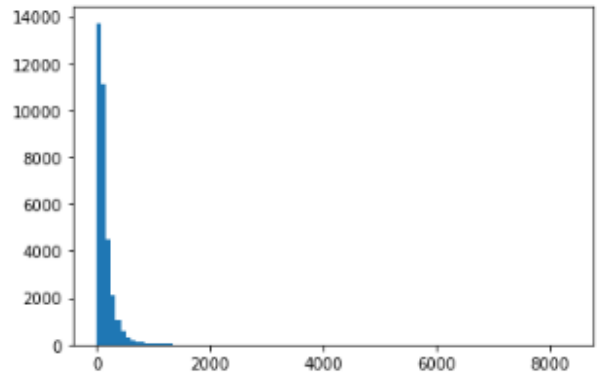
Distribution of polarity:



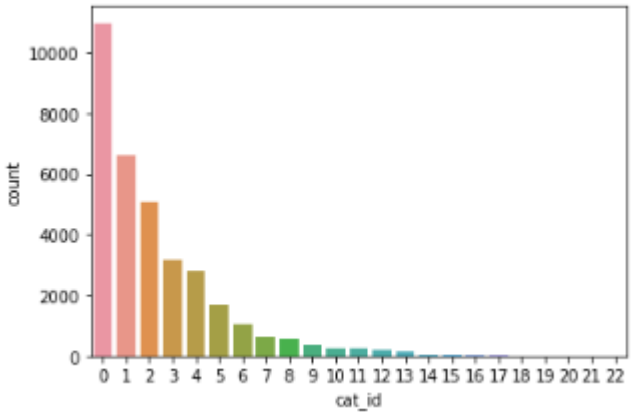
Distribution of review rating:



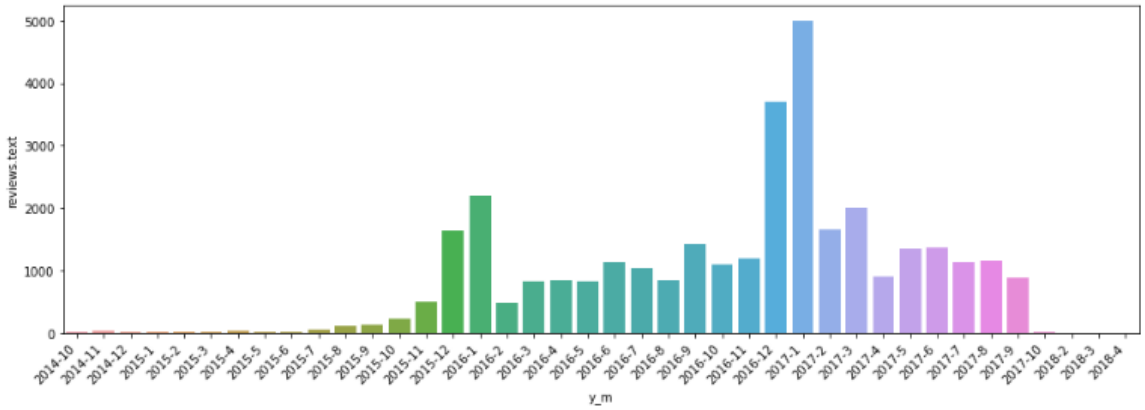
Distribution of review len:



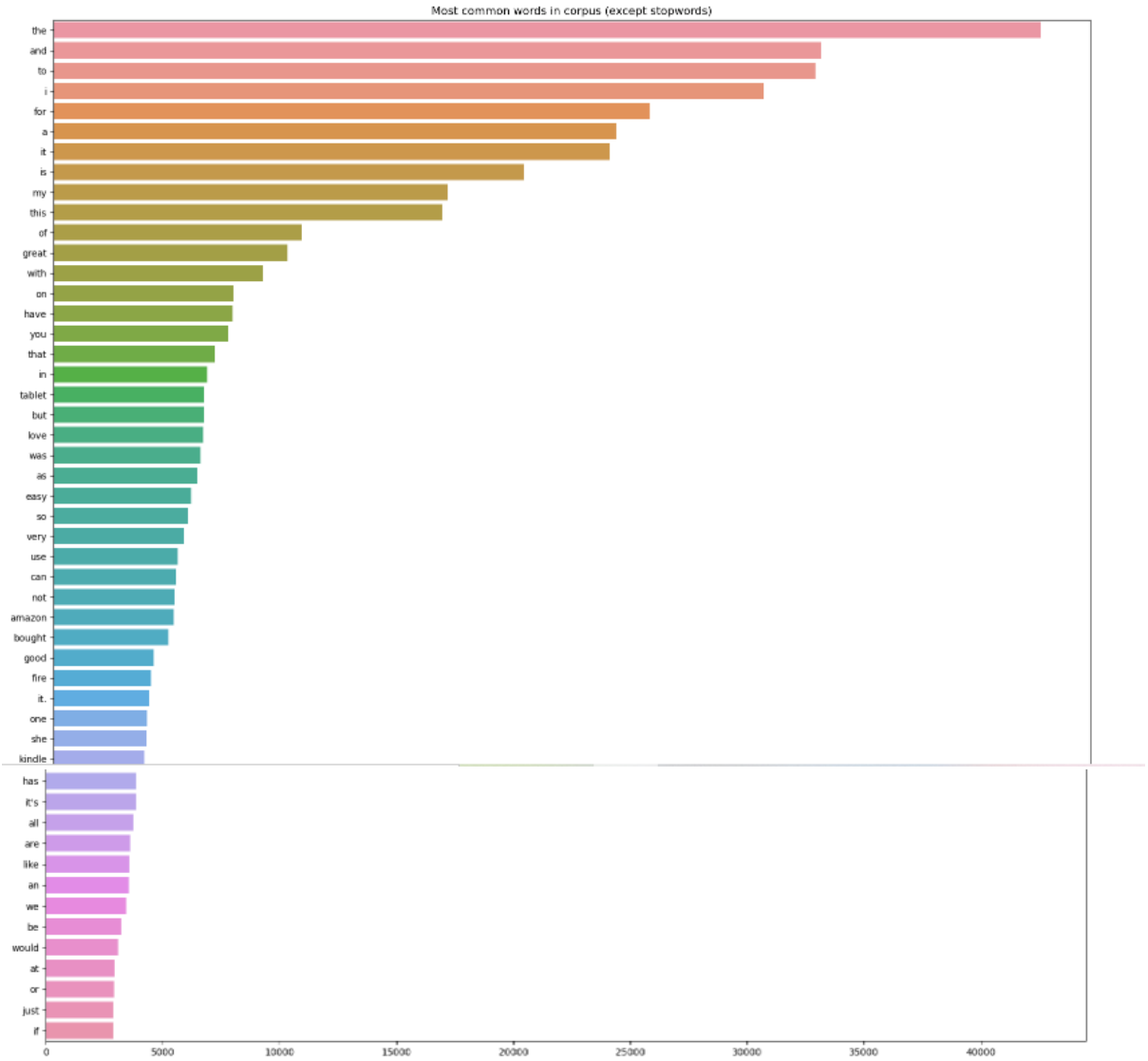
Distribution of product categories:



Distribution of month of reviews:

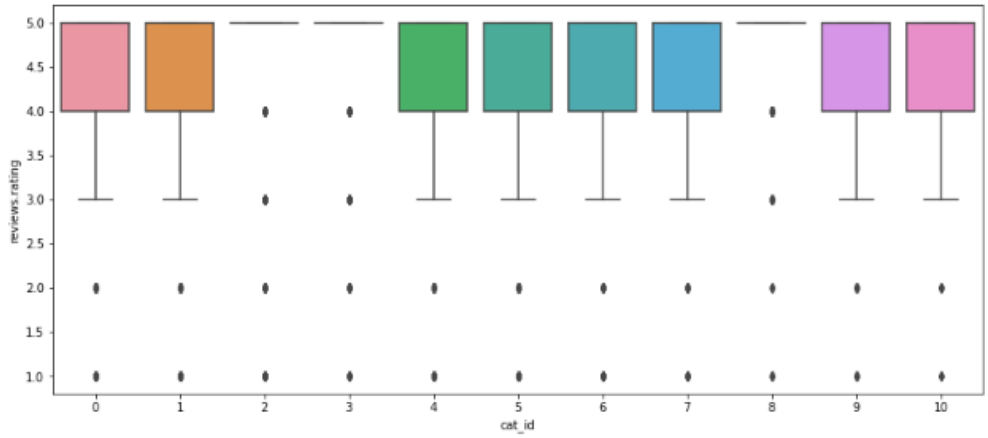


Distribution of most common words: found that the word” THE “ has most usage.

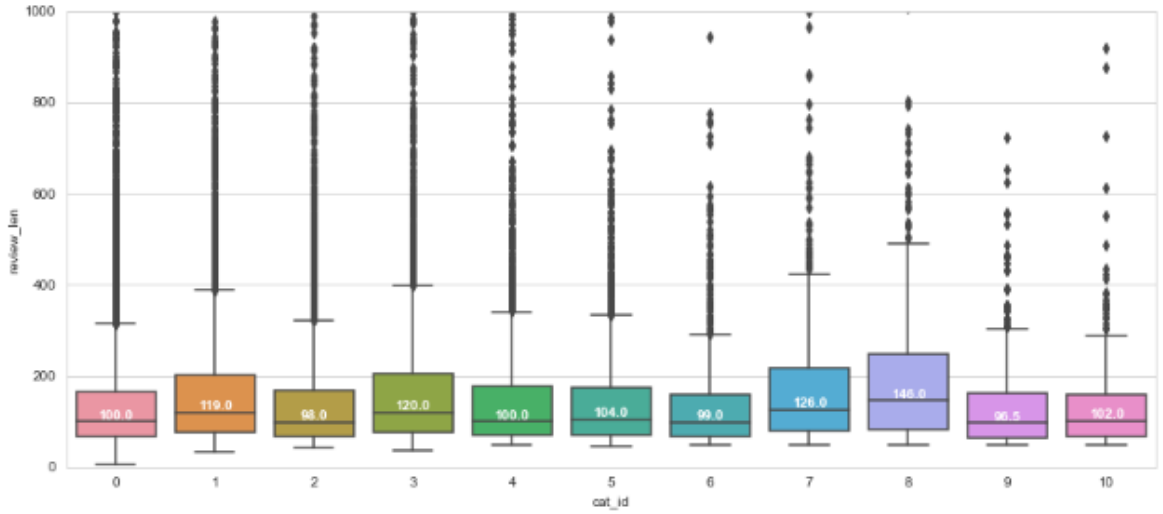


BOX PLOT:

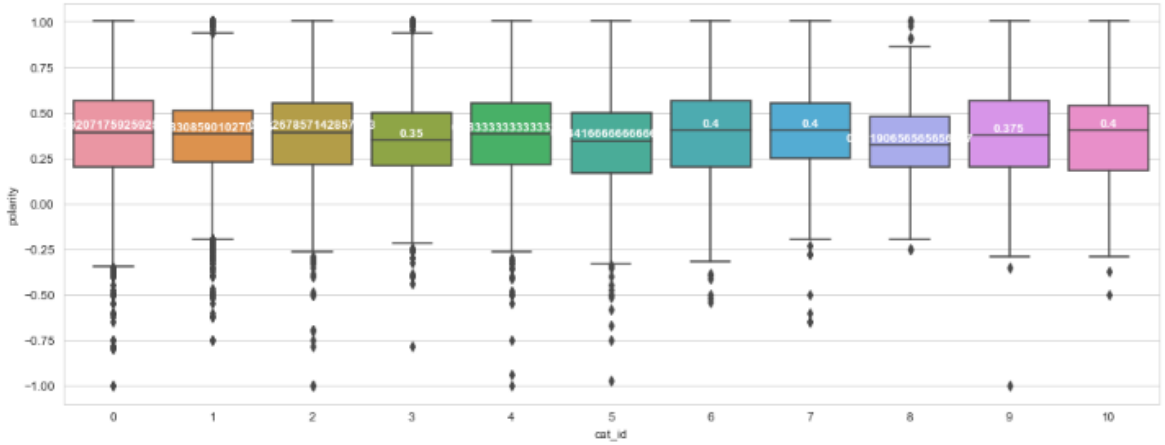
DISTRIBUTION FOR RATING ACROSS CATEGORIES:



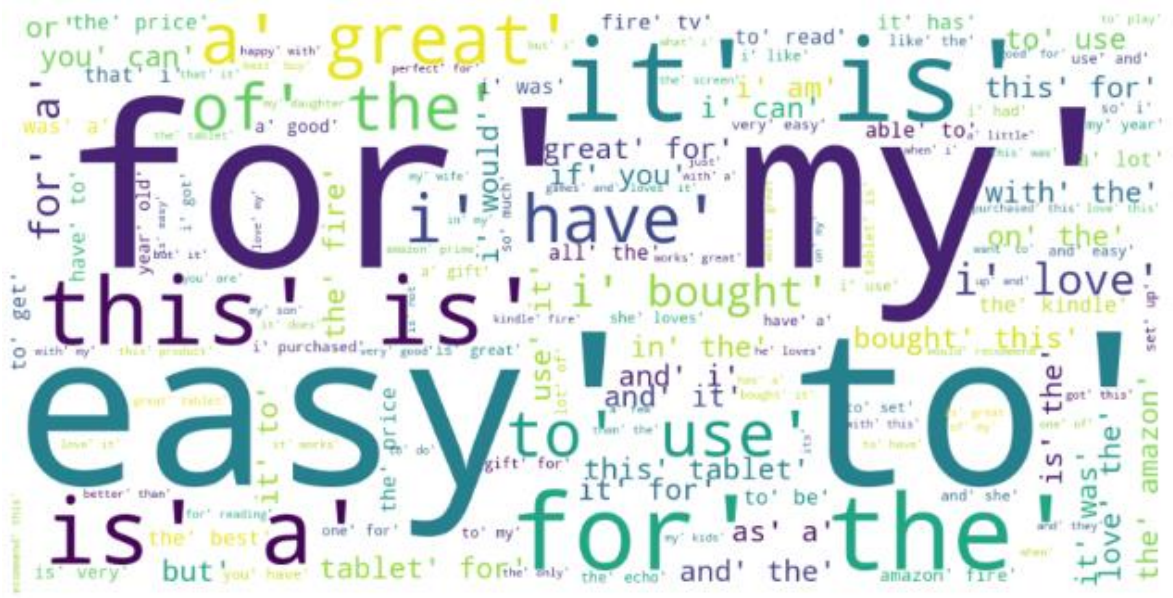
DISTRIBUTION OF REVIEW LENGTH ACROSS CATEGORIES:



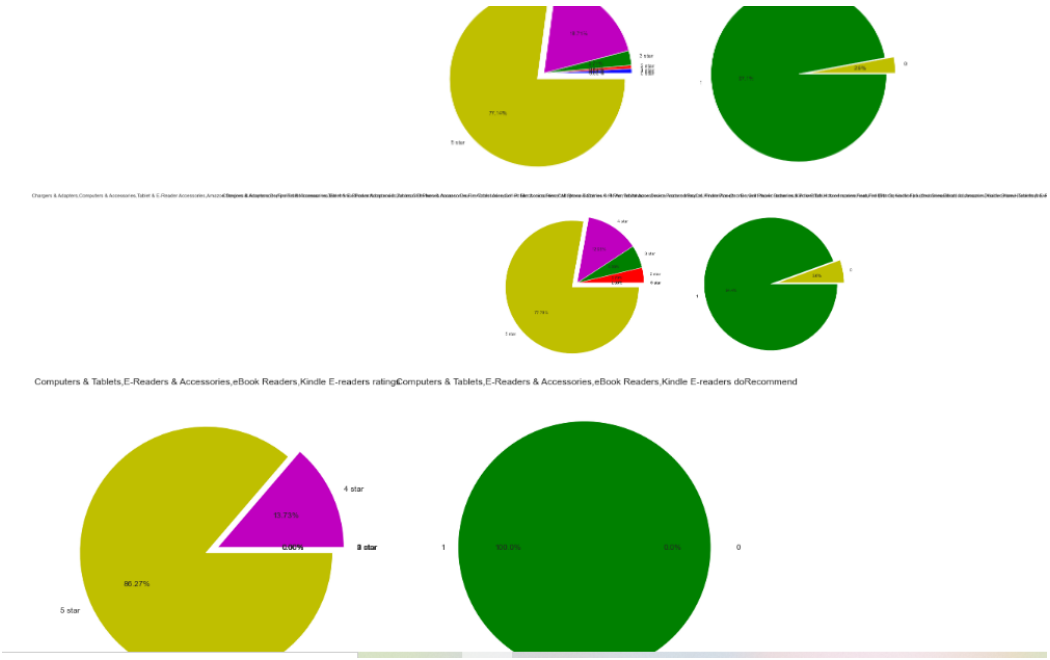
DISTRIBUTION OF SENTIMENT SCORE ACROSS CATEGORIES:



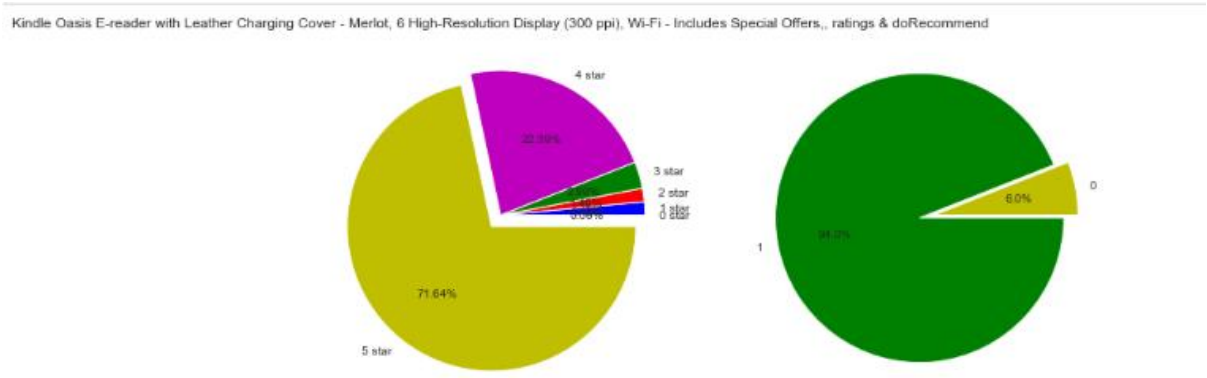
WORDCLOUD:



PIE CHART WHEN WE GROUP BY CATEGORIES& RATING:



PIE CHART WHEN WE GROUP BY NAME AND RATING:



STEP 6: BUILDING MODEL USING NLTK AND STOP WORDS.

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
STOPWORDS = set(stopwords.words('english'))
```

```
data['reviews.text'] = data['reviews.text'].apply(lambda x: ' '.join([w for
w in x.split(' ') if (w not in STOPWORDS)]))
```

```
review=pd.DataFrame(data.groupby('reviews.rating').size().sort_values(ascen
ding=False).rename('No of Users').reset_index())
review.head()
```

Out[42]:

	reviews.rating	No of Users
0	5.0	48
1	4.0	15
2	3.0	2
3	2.0	1
4	1.0	1

```
permanent = data[['reviews.rating' , 'reviews.text' , 'reviews.title' ,
'reviews.username']]
mpermanent=permanent.dropna()
mpermanent.head()
```

Out[44]:

	reviews.rating	reviews.text	reviews.title	reviews.username
2814	5.0	lightweight portable excellent battery life.	works great	Purchaser1
2815	5.0	like much voyage. shape makes easier holding. ...	in love	kcladyz
2816	5.0	replacing older reader without light traveling...	great size	Bbshop
2817	4.0	first e-reader. know odd refresh took little g...	very light	diannez
2818	4.0	kindle awesome. love design it. nothing softwa...	great kindle	Brandon

```
check =  mpermanent[mpermanent["reviews.text"].isnull()]
check.head()
actualrating = mpermanent[(mpermanent['reviews.rating'] == 1) |
(mpermanent['reviews.rating'] == 5)]
actualrating.shape
```

```
y = actualrating['reviews.rating']
x = actualrating['reviews.text'].reset_index()
# X =x[xindex(False)]
```

```
import string
from nltk.corpus import stopwords
# stop=set(stopwords.words('english'))
def text_process(text):
    '''
    Takes in a string of text, then performs the following:
    1. Remove all punctuation
    2. Remove all stopwords
    3. Return the cleaned text as a list of words
    '''
    nopunc = [char for char in text if char not in string.punctuation]
    nopunc = ''.join(nopunc)

    return [word for word in nopunc.split() if word.lower() not in
stopwords.words('english')]
```

```
nltk.download('stopwords')
sample_text = "Hey there! This project is done by Shivani, Deepika and
Harini." #TOKENIZATION
print(text_process(sample_text))
```

```
In [54]: nltk.download('stopwords')
         sample_text = "Hey there! This project is done by Shivani, Deepika and Harini."
         print(text_process(sample_text))

         ['Hey', 'project', 'done', 'Shivani', 'Deepika', 'Harini']
```

USING BOW TRANSFORMER:

```
from sklearn.feature_extraction.text import CountVectorizer
# next we need to vectorize our input variable (X)
#we use the count vectoriser function and the analyser we use is the above
lines of code
# this should return a vector array
bow_transformer = CountVectorizer(analyzer=text_process).fit(X)
```

```
len(bow_transformer.vocabulary_)
```

STEP 7: TRAINING THE MODEL AND SPLITTING THEM INTO TRAINING AND TESTING:

```
#Lets start training the model
from sklearn.model_selection import train_test_split
```

```
#using 30% of the data for testing, this will be revised once we do not get
the desired accuracy
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=101)


#we need to have a label for
# lets have a label which group the stars into two groups, 1 for good, 0
for bad
# so anything more than 3 , 3 being neutral is good, rest bad
# data['label'] = ['1' if reviews.rating > 3 else '0' for reviews.rating in
data['reviews.rating']];
mpermanent['label'] = ['1' if star >= 3 else '0' for star in
mpermanent['reviews.rating']];
```

In [66]:
```
mpermanent
```

Out[66]:

	reviews.rating	reviews.text	reviews.title	reviews.username	label
2814	5.0	lightweight portable excellent battery life.	works great	Purchaser1	1
2815	5.0	like much voyage. shape makes easier holding. ...	in love	kcladyz	1
2816	5.0	replacing older reader without light traveling...	great size	Bbshop	1
2817	4.0	first e-reader. know odd refresh took little g...	very light	diannez	1
2818	4.0	kindle awesome. love design it. nothing softwa...	great kindle	Brandon	1
...	...	...	...	...	...
2876	4.0	looking forward bought right popped available ...	fantastic, if you can afford it	Nitin	1
2877	5.0	used kindle e-readers since amazon introduced ...	the reviews are right-this is the best kindle ...	rlmgolfer	1
2878	5.0	kindle easily best one i've ever had. comfort...	best kindle yet	somileo	1
2879	5.0	best kindle ereader yet. size weight make perf...	best kindle yet	RobDrob	1
2880	5.0	actually easier hold paperback, might even lig...	the lightest ereader by far!	FloridaOleFart	1

67 rows × 5 columns

```
stop = set(stopwords.words('english'))


def clean_document(doco):
    punctuation = string.punctuation
    punc_replace = ''.join([' ' for s in punctuation])
    doco_link_clean = re.sub(r'http\S+', '', doco)
    doco_clean_and = re.sub(r'&\S+', '', doco_link_clean)
    doco_clean_at = re.sub(r'@\S+', '', doco_clean_and)
    doco_clean = doco_clean_at.replace('-', ' ')
    doco_alphas = re.sub(r'\W +', ' ', doco_clean)
    trans_table = str.maketrans(punctuation, punc_replace)
    doco_clean = ' '.join([word.translate(trans_table) for word in
doco_alphas.split(' ')])
    doco_clean = doco_clean.split(' ')
    p = re.compile(r'\s*\b(?=[a-z\d]*([a-z\d])\1{3}|\d+\b)[a-z\d]+',
re.IGNORECASE)
    doco_clean = ([p.sub("", x).strip() for x in doco_clean])
    doco_clean = [word.lower() for word in doco_clean if len(word) > 2]
    doco_clean = ([i for i in doco_clean if i not in stop])
#     doco_clean = [spell(word) for word in doco_clean]
#     p = re.compile(r'\s*\b(?=[a-z\d]*([a-z\d])\1{3}|\d+\b)[a-z\d]+',
re.IGNORECASE)
    doco_clean = ([p.sub("", x).strip() for x in doco_clean])
```

```
#     doco_clean = ([spell(k) for k in doco_clean])
    return doco_clean


# Generate a cleaned reviews array from original review texts
review_cleans = [clean_document(doc) for doc in reviews];
sentences = [' '.join(r) for r in review_cleans ]


#Keras
tokenizer = Tokenizer()
tokenizer.fit_on_texts(sentences)


text_sequences = np.array(tokenizer.texts_to_sequences(sentences))
sequence_dict = tokenizer.word_index
word_dict = dict((num, val) for (val, num) in sequence_dict.items())


reviews_encoded = [];
for i,review in enumerate(review_cleans):
    reviews_encoded.append([sequence_dict[x] for x in review]);


max_cap =8;
X = pad_sequences(reviews_encoded, maxlen=max_cap, truncating='post')
Y = np.array([[0,1] if '0' in label else [1,0] for label in labels])
np.random.seed(1024);
random_posits = np.arange(len(X))
np.random.shuffle(random_posits);

X = X[random_posits];
Y = Y[random_posits];

train_cap = int(0.85 * len(X));
dev_cap = int(0.93 * len(X));

X_train, Y_train = X[:train_cap], Y[:train_cap]
X_dev, Y_dev = X[train_cap:dev_cap], Y[train_cap:dev_cap]
X_test1, Y_test1 = X[dev_cap:], Y[dev_cap:]
```

STEP 8: LSTM MODEL

```
from keras.models import Sequential
from keras.layers import Dense
from keras.backend import eval

from keras.layers import LSTM
from keras.layers.embeddings import Embedding


model1 = Sequential();
model1.add(Embedding(len(word_dict)+1, max_cap, input_length=max_cap));
#adding a LSTM layer of dim 1--
model1.add(LSTM(150, return_sequences=True));
model1.add(LSTM(150, return_sequences=False));
```

```
#adding a dense layer with activation function of relu
model1.add(Dense(100, activation='relu'));#best 50,relu
#adding the final output activation with activation function of softmax
model1.add(Dense(2, activation='sigmoid'));
print(model1.summary());
optimizer = Adam(lr=0.0001, decay=0.0001);

model1.compile(loss='binary_crossentropy', optimizer=optimizer,
metrics=['accuracy'])
# fit model and run it for 5 epochs
history = model1.fit(X_train, Y_train, batch_size=16, epochs=5,
validation_data=(X_dev, Y_dev))
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 8, 8)              106888

 lstm_4 (LSTM)               (None, 8, 150)            95400

 lstm_5 (LSTM)               (None, 150)               180600

 dense_4 (Dense)             (None, 100)               15100

 dense_5 (Dense)             (None, 2)                 202

=================================================================
Total params: 398,190
Trainable params: 398,190
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/5
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/adam.py:105: UserWarning: The `lr` argument is deprecated, use `learning_rate` instea
  super(Adam, self).__init__(name, **kwargs)
1840/1840 [==============================] - 159s 84ms/step - loss: 0.1266 - accuracy: 0.9763 - val_loss: 0.0997 - val_accuracy: 0.9769
Epoch 2/5
1840/1840 [==============================] - 130s 71ms/step - loss: 0.0879 - accuracy: 0.9767 - val_loss: 0.0893 - val_accuracy: 0.9769
Epoch 3/5
1840/1840 [==============================] - 103s 56ms/step - loss: 0.0777 - accuracy: 0.9772 - val_loss: 0.0924 - val_accuracy: 0.9772
Epoch 4/5
1840/1840 [==============================] - 132s 72ms/step - loss: 0.0720 - accuracy: 0.9780 - val_loss: 0.0942 - val_accuracy: 0.9772
Epoch 5/5
1840/1840 [==============================] - 149s 81ms/step - loss: 0.0672 - accuracy: 0.9791 - val_loss: 0.0960 - val_accuracy: 0.9769
```

MODEL EVALUATION:

In [103…

```
score = model1.evaluate(X_test1, Y_test1)
print("Test accuracy: %0.4f%%" % (score[1]*100))
```

```
76/76 [==============================] - 1s 9ms/step - loss: 0.1052 - accuracy: 0.9732
Test accuracy: 97.3185%
```

We have achieved 97.3185% accuracy through LSTM model.

CONCLUSION:

In conclusion, with this study, I tried to show how sentiment analysis works by applying it on Amazon review data. As the results on the test data shows, LSTM networks are the most suitable for binary sentiment analysis on Amazon.com product reviews.

REFERENCE:

1) Basilico, Justin & Hofmann, Thomas. (2004). Unifying Collaborative and Content-Based Filtering. Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004. 10.1145/1015330.1015394.

2) Meteren, Robin. (2000). Using Content-Based Filtering for Recommendation.

3) Bharadwaj, Dr. (2019). THE WAR BETWEEN FLIPKART AND AMAZON INDIA: A STUDY ON CUSTOMER PERCEPTION.

4) P. M. Alamdari, N. J. Navimipour, M. Hosseinzadeh, A. A. Safaei and A. Darwesh, "A Systematic Study on the Recommender Systems in the E-Commerce," in IEEE Access, vol. 8, pp. 115694-115716, 2020, doi: 10.1109/ACCESS.2020.3002803.

5) Alamdari, Pegah & Navimipour, Nima & Hosseinzadeh, Mehdi & Safaei, Ali & Darwesh, Aso. (2020). A Systematic Study on the Recommender Systems in the E-Commerce. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.3002803.

6) D.N.V.Krishna Reddy, D. R. (2015). A Study On Customer's Perception And Satisfaction Towards Electronic Banking In Khammam District. IOSR Journal of Business and Management (IOSR-JBM), 17 (12 (II)), 20-27.

7) Dahiya, R. (2012). Impact of Demographic Factors of Consumers on Online Shopping Behaviour: A Study of Consumers in India. International Journal of Engineering and Management Sciences, 3 (1), 43-52