TOPIC MODELING ON AMAZON REVIEWS

NLP Project Report - 2021

Submitted by

Dara Sravan Kumar (2017IMT-030)

submitted to

Dr. Debanjan Sadhya

ABV INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT GWALIOR

2021

TABLE OF CONTENTS

Chapter-1

Introduction & Objectives

1.1 Introduction

The present-day applications contain a large amount of text data that is almost impossible for humans to read and preprocess. The user reviews on the world wide web are one such data which are millions in number. According to statistics, almost 1.2 million reviews on amazon itself were posted from 1995 to 2013 and are still increasing. Due to the increase of large amounts of text data, modern-day solutions are implemented using Natural Language Processing. One of such modern-day solutions is topic modeling. Topic modeling is a natural language processing technique used to discover topics by analyzing the text data in documents and finding the hidden patterns. This can be used in many ways as it has many advantages. One such example is mining user reviews of a product to categorize them based on their topics. This helps the users know the pros and cons of a product and also helps the merchants realize in what areas the product needs to be updated.

Modern applications like Amazon, Flipkart, Myntra, etc., use the feature mentioned above. These applications contain many user reviews after the popularity of a product is decided by the user reviews. Topic modeling is also used in many ways, like fetching information from unstructured text data, document clustering, and feature selection. The topic modeling is needed in shopping websites and other websites like Wikipedia, which contains millions of documents. These days the topic modeling is also used in the news field to fetch trending topics on the internet. In this project, we are using amazon reviews of a car-based product as text data. The project aims to predict the most definite number of group words from the user reviews. This group of words represents the topics themselves.

1.2 Problem Statement & Objectives

The project aims to demonstrate how to build a model that can achieve the task of grouping user reviews where each group represents a topic using the concept of topic modeling. In our case, we are using amazon's automotive review data. Precisely, we are building a model that explicitly gives the details and rating about significant product elements by using the user reviews and user ratings (in our case, car products). For example, In the reviews section of any mobile phone on amazon's page contains specific topics like display, battery, etc., along with ratings for each topic. Our project aims to build such a model for automotive review data.

Following are the objectives which describe our project precisely:

- Extracting and Naming the topics from the dataset
- Grouping the reviews based on their topics
- Rating each topic based on the review's rating
- Providing Sentiments on each topic.
- To ensure maximum customer satisfaction and ready to make decisions.
- To help the seller realize in what areas the product needs to be updated.

Chapter 2

Design Details & Implementation.

2.1 Design Methodology

Topic Modelling on Amazon Reviews contains a sequence of steps and methods involved, and also we have specific preprocessing techniques for text data. Following Fig 2.1 shows the detailed design methodology of our project.

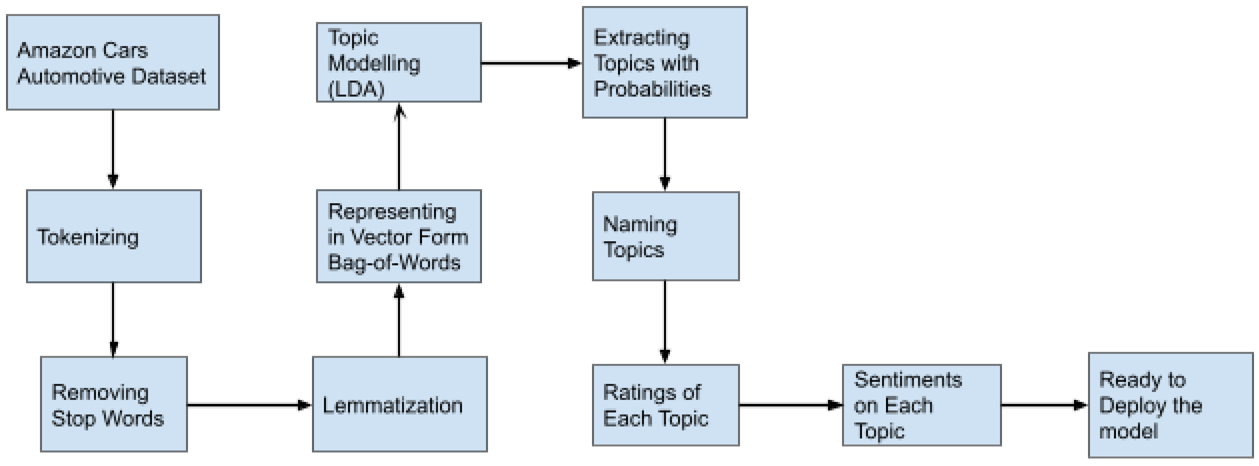

Amazon Cars Automotive Dataset
Tokenizing
Removing Stop Words
Lemmatization
Representing in Vector Form Bag-of-Words
Topic Modelling (LDA)
Extracting Topics with Probabilities
Naming Topics
Ratings of Each Topic
Sentiments on Each Topic
Ready to Deploy the model

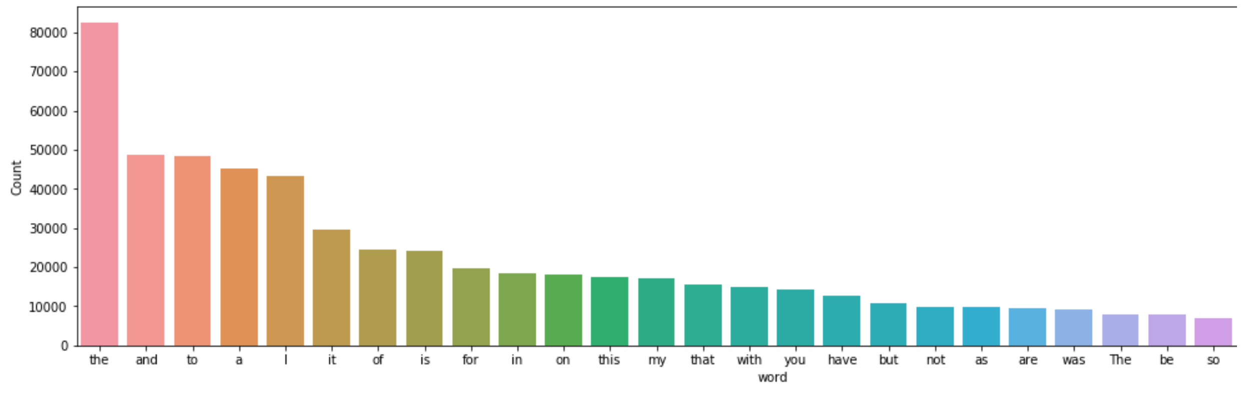
Fig 2.1 Design Methodology

2.2 Stop Words Removal

“Stop Words” in-text data are common words that are often used for expressing or communicating. Stop Words in NLP are used to analyze the text data and extract valuable insights from text data. Examples of stop words are: is, are, being, although, etc. Stop Word Removal is a vital tool in text preprocessing and language modeling. Stop words are removed from text to understand the exact meaning of the text data. By removing these words, we can address the computational constraint of NLP as removing these stop words decreases data size and improves the efficiency of language modeling. Some applications need to remove these stop

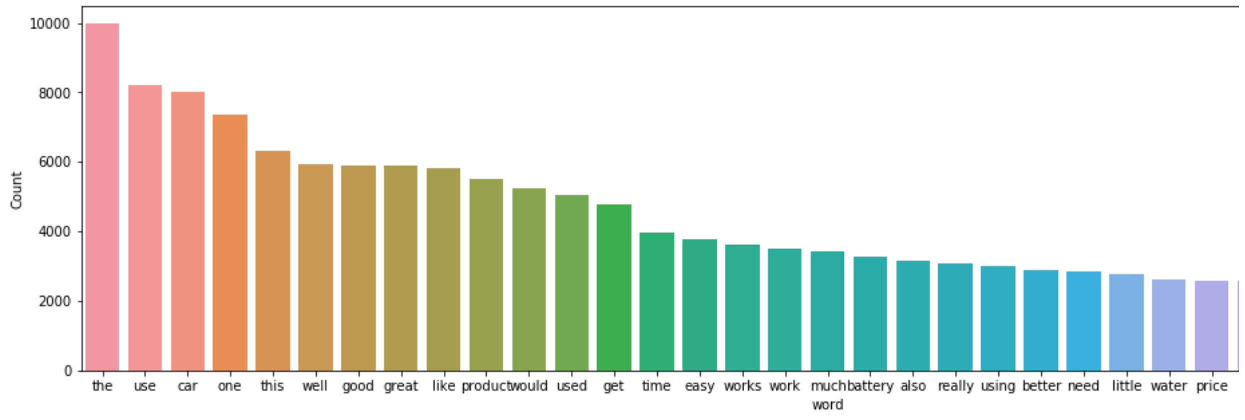
words, whereas some applications need to persist the stop words. Therefore removal of these stop words depends on our application where we are using it in NLP.
When we plot the frequency of words in our amazon reviews dataset, We can see many stop words in the customer reviews, which are meaningless.

```
freq_words(df['reviewText'])
```



We have removed stop words to understand the customer reviews. Every customer has a specific way of expressing their reviews, so to understand the exact meaning of customer reviews, we have removed stop words.

```
freq_words(reviews, 35)
```



We can also see the frequency plot after removing the stopwords in the above figure. Now we can work on the meaningful text data to generate useful insights.

2.3 Tokenization

Tokenization is an important task in text preprocessing techniques of Natural Language Processing. Tokenization helps to break the strings of text into words,

phrases, etc. Which are called Tokens. These Tokens are very helpful in further processing of data into language models for NLP tasks. Tokenization helps in understanding the meaning of the text data by breaking it into tokens.
Generally, Word Tokenization divides the text into individual words. Whereas, Sentence Tokenization divides the text into individual sentences. Both methods are useful in text preprocessing, It helps in understanding the meaning of the sentence for better language modeling in NLP.

```
tokenized_reviews = pd.Series(reviews).apply(lambda x: x.split())
print(tokenized_reviews[1])
```

```
['these', 'long', 'cables', 'work', 'fine', 'truck', 'quality', 'seems', 'little', 'shabby', 'side', 'for',
'cables', 'seem', 'like', 'would', 'see', 'chinese', 'knock', 'shop', 'like', 'harbor', 'freight', 'bucks']
```

```
print(tokenized_reviews[1])
len(tokenized_reviews[1])
```

```
['these', 'long', 'cables', 'work', 'fine', 'truck', 'quality', 'seems', 'little', 'shabby', 'side', 'for',
'cables', 'seem', 'like', 'would', 'see', 'chinese', 'knock', 'shop', 'like', 'harbor', 'freight', 'bucks']

29
```

We can see the tokens after performing tokenization on the amazon reviews dataset. We have used the tokenization technique on amazon reviews data to break the strings into words to work efficiently on further processing of data.

2.4 Lemmatization

In Natural Language Processing, lemmatization is considered one of the most essential text pre-processing techniques in order to extract the context of the text data. Lemmatization is an iterative process of removing inflectional forms such as suffixes, prefixes, etc. In a linguistic sense, the term inflection refers to a process of word-formation that is formed by changing the grammatical form of the main word. The grammatical forms include case gender, animacy, mood, voice, aspect, person, number, tense, and definiteness. In simpler terms, in lemmatization, we try to reduce the different forms of a word to their root form.

```
reviews_2 = lemmatization(tokenized_reviews)
print(reviews_2[1])
```

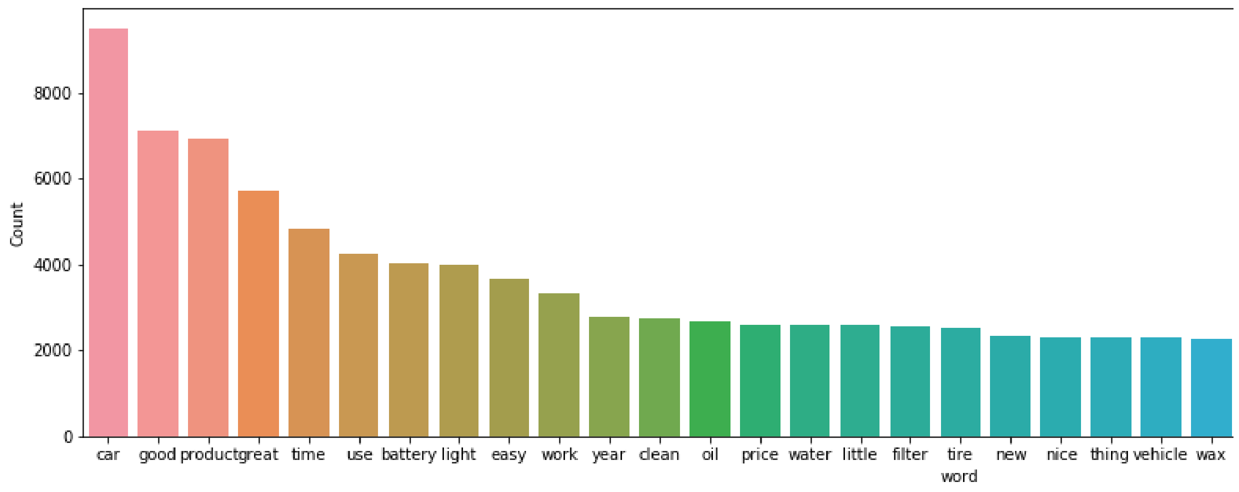
```
['long', 'cable', 'fine', 'truck', 'quality', 'little', 'shabby', 'side', 'money', 'dollar', 'jumper', 'cable', 'chinese',
k']
```

```
print(reviews_2[1])
len(reviews_2[1])
```

```
['long', 'cable', 'fine', 'truck', 'quality', 'little', 'shabby', 'side', 'money', 'dollar', 'jumper', 'cable', 'chinese',
k']
17
```

Lemmatization converts the word to its nearest word available in the dictionary, then searches for another word in the dictionary, This process is continued iteratively until no word is present further. We can see the lemmas of the words in the above figure which gives us the exact context of the customer reviews.

```
freq_words(df['reviews'], 35)
```



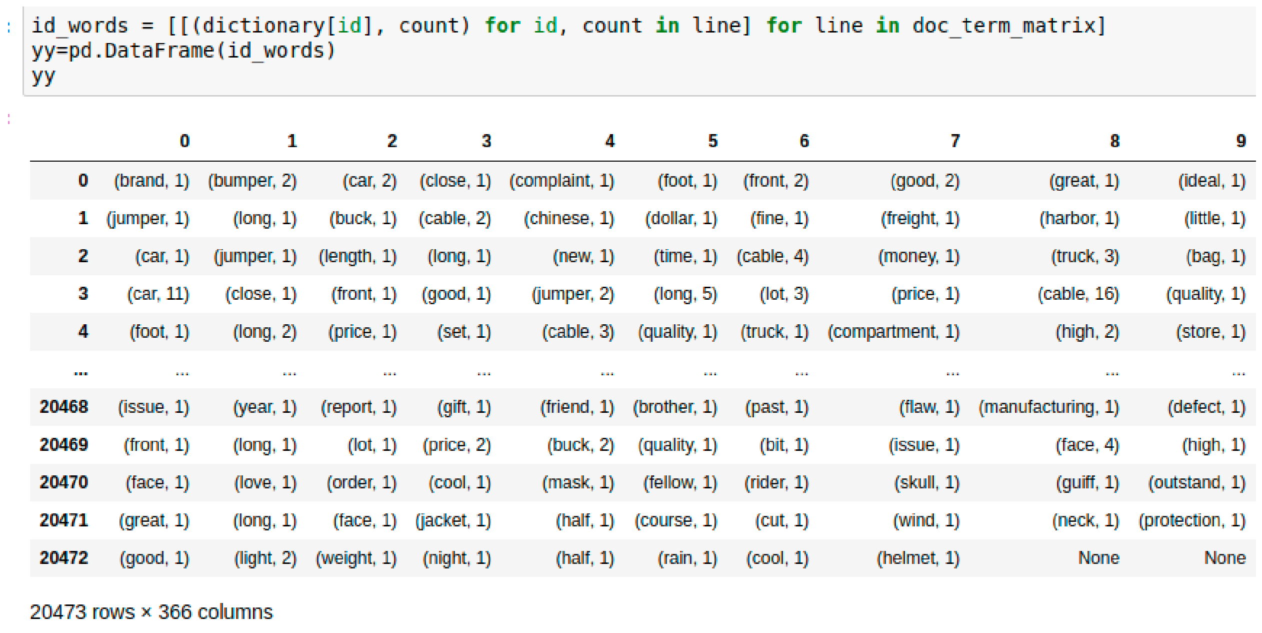
When we plot the frequency of the words after preprocessing the data with essential tools, techniques, and methods. We can see that there is much helpful information available to extract topics and sentiments from the data.

2.5 Bag of Words(BoW)

Our text cannot be feed into this algorithm directly. It works with numbers and the machine does not understand text data. Thus, the Words bag model is used to preprocess text in vector format, which continues to compute the total occurrences of the words most commonly used. The Bag of Words is a technology to convert text phrases into numerical vectors. The Bag of Words model (BoW) is the most simple form of displaying text in form of numbers.

We first create a dictionary of all the unique words in the above amazon automotive customer reviews dataset. We assign an id to the unique words and try to display them in the form of vectors using the event count.

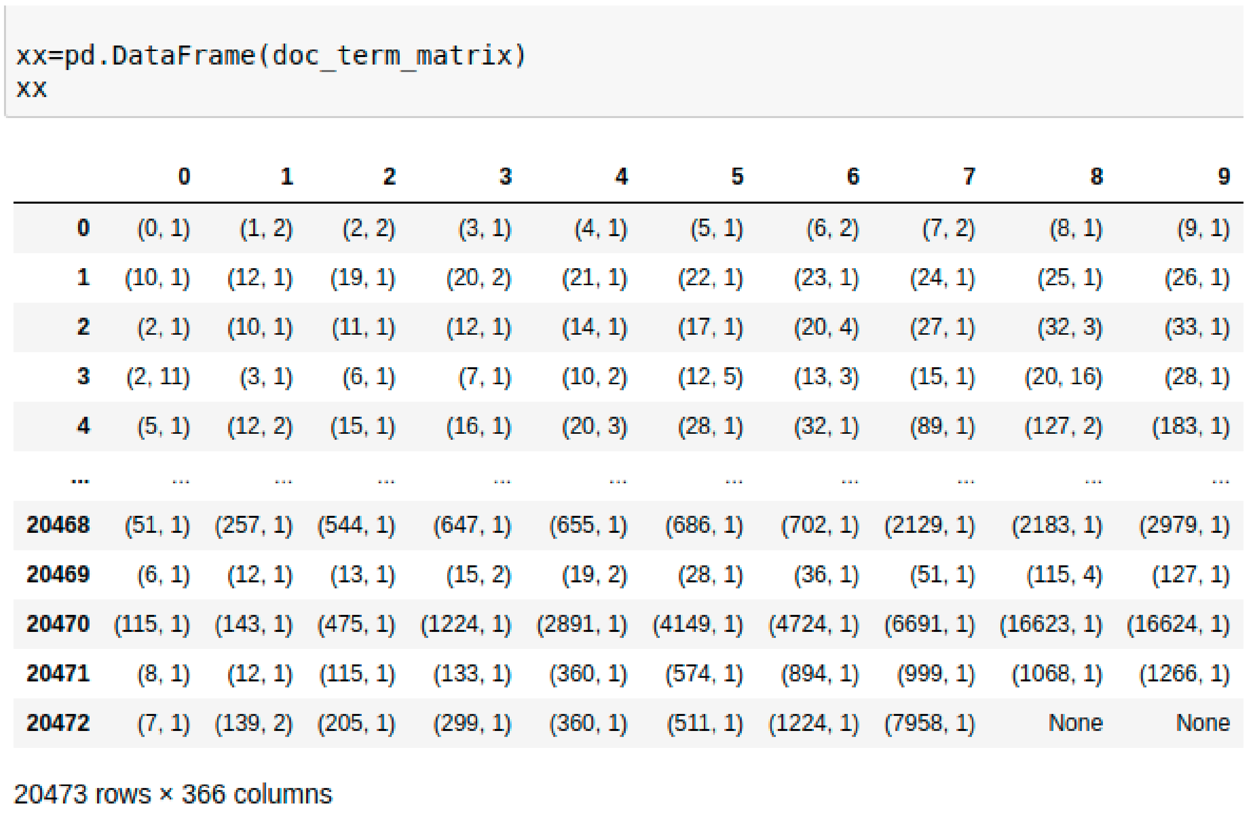
The figure below shows clearly the BoW representation. An ID and its frequency are assigned to every single word to represent the occurrence of the word in the matrix. We haven't shown one with words with zero frequency.



```
id_words = [[(dictionary[id], count) for id, count in line] for line in doc_term_matrix]
yy=pd.DataFrame(id_words)
yy
```

	0	1	2	3	4	5	6	7	8	9
0	(brand, 1)	(bumper, 2)	(car, 2)	(close, 1)	(complaint, 1)	(foot, 1)	(front, 2)	(good, 2)	(great, 1)	(ideal, 1)
1	(jumper, 1)	(long, 1)	(buck, 1)	(cable, 2)	(chinese, 1)	(dollar, 1)	(fine, 1)	(freight, 1)	(harbor, 1)	(little, 1)
2	(car, 1)	(jumper, 1)	(length, 1)	(long, 1)	(new, 1)	(time, 1)	(cable, 4)	(money, 1)	(truck, 3)	(bag, 1)
3	(car, 11)	(close, 1)	(front, 1)	(good, 1)	(jumper, 2)	(long, 5)	(lot, 3)	(price, 1)	(cable, 16)	(quality, 1)
4	(foot, 1)	(long, 2)	(price, 1)	(set, 1)	(cable, 3)	(quality, 1)	(truck, 1)	(compartment, 1)	(high, 2)	(store, 1)
...	...	...	...	...	...	...	...	...	...	...
20468	(issue, 1)	(year, 1)	(report, 1)	(gift, 1)	(friend, 1)	(brother, 1)	(past, 1)	(flaw, 1)	(manufacturing, 1)	(defect, 1)
20469	(front, 1)	(long, 1)	(lot, 1)	(price, 2)	(buck, 2)	(quality, 1)	(bit, 1)	(issue, 1)	(face, 4)	(high, 1)
20470	(face, 1)	(love, 1)	(order, 1)	(cool, 1)	(mask, 1)	(fellow, 1)	(rider, 1)	(skull, 1)	(guiff, 1)	(outstand, 1)
20471	(great, 1)	(long, 1)	(face, 1)	(jacket, 1)	(half, 1)	(course, 1)	(cut, 1)	(wind, 1)	(neck, 1)	(protection, 1)
20472	(good, 1)	(light, 2)	(weight, 1)	(night, 1)	(half, 1)	(rain, 1)	(cool, 1)	(helmet, 1)	None	None

20473 rows × 366 columns



```
xx=pd.DataFrame(doc_term_matrix)
xx
```

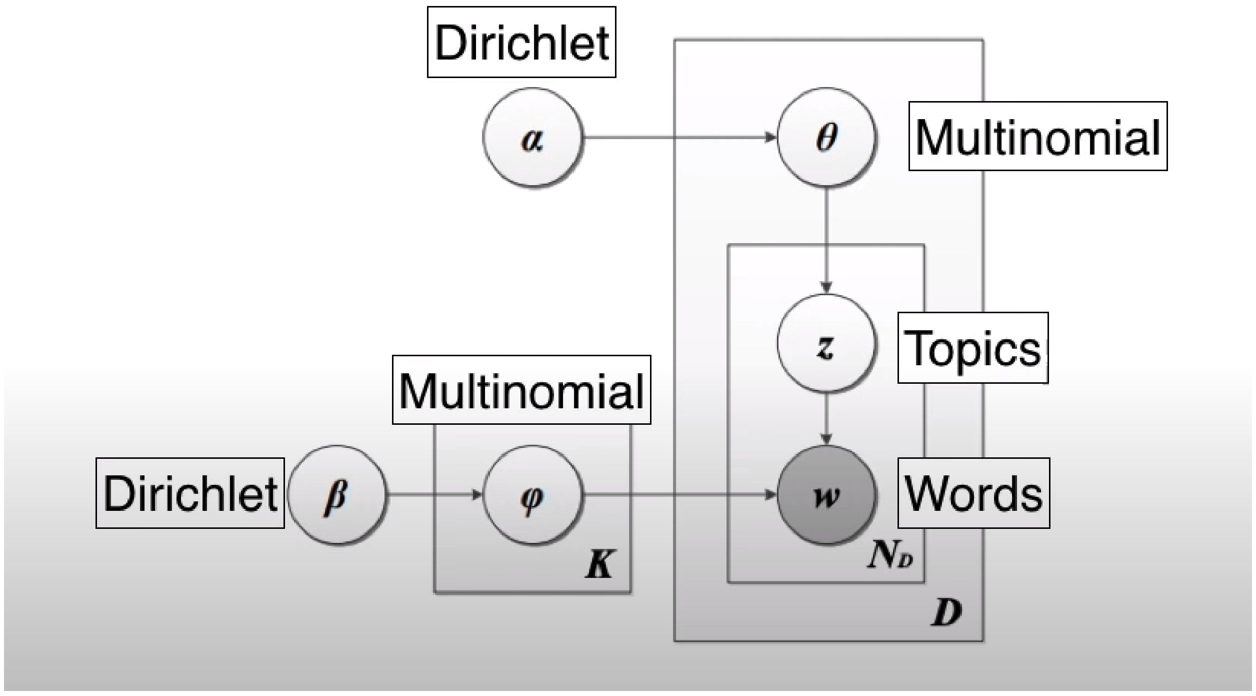
	0	1	2	3	4	5	6	7	8	9
0	(0, 1)	(1, 2)	(2, 2)	(3, 1)	(4, 1)	(5, 1)	(6, 2)	(7, 2)	(8, 1)	(9, 1)
1	(10, 1)	(12, 1)	(19, 1)	(20, 2)	(21, 1)	(22, 1)	(23, 1)	(24, 1)	(25, 1)	(26, 1)
2	(2, 1)	(10, 1)	(11, 1)	(12, 1)	(14, 1)	(17, 1)	(20, 4)	(27, 1)	(32, 3)	(33, 1)
3	(2, 11)	(3, 1)	(6, 1)	(7, 1)	(10, 2)	(12, 5)	(13, 3)	(15, 1)	(20, 16)	(28, 1)
4	(5, 1)	(12, 2)	(15, 1)	(16, 1)	(20, 3)	(28, 1)	(32, 1)	(89, 1)	(127, 2)	(183, 1)
...	...	...	...	...	...	...	...	...	...	...
20468	(51, 1)	(257, 1)	(544, 1)	(647, 1)	(655, 1)	(686, 1)	(702, 1)	(2129, 1)	(2183, 1)	(2979, 1)
20469	(6, 1)	(12, 1)	(13, 1)	(15, 2)	(19, 2)	(28, 1)	(36, 1)	(51, 1)	(115, 4)	(127, 1)
20470	(115, 1)	(143, 1)	(475, 1)	(1224, 1)	(2891, 1)	(4149, 1)	(4724, 1)	(6691, 1)	(16623, 1)	(16624, 1)
20471	(8, 1)	(12, 1)	(115, 1)	(133, 1)	(360, 1)	(574, 1)	(894, 1)	(999, 1)	(1068, 1)	(1266, 1)
20472	(7, 1)	(139, 2)	(205, 1)	(299, 1)	(360, 1)	(511, 1)	(1224, 1)	(7958, 1)	None	None

20473 rows × 366 columns

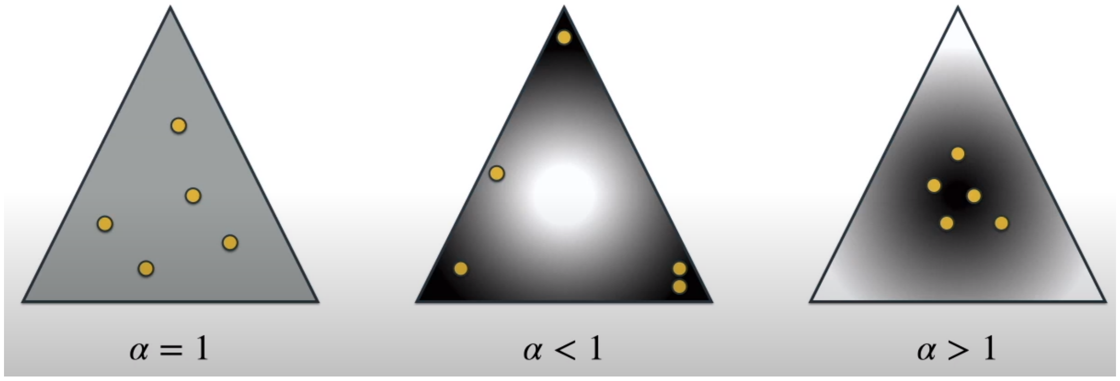
We can see from the above figure that our whole amazon text data after preprocessing is represented in the form of vectors that contains an id along with its frequency in the matrix representation. There are a lot of models to convert our data into vector forms such as TF-IDF, Inverse-TF-IDF, etc. Our area of application is to find out topics using topic modeling which requires a significantly less amount of information, unlike machine learning applications. So we went with a simpler model to avoid sparse matrices which are computationally expensive.

2.6 LDA (Latent Dirichlet Allocation)

LDA is a kind of statistically unsupervised model for the classification into their respective topics of the document collection.

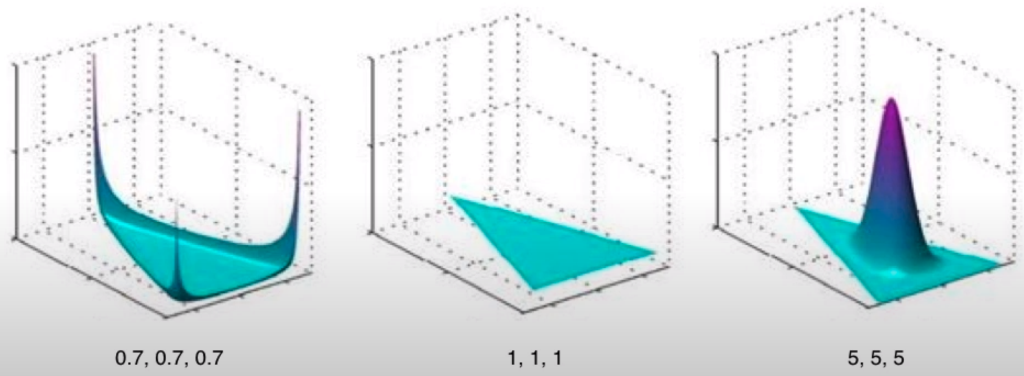


Dirichlet Distributions



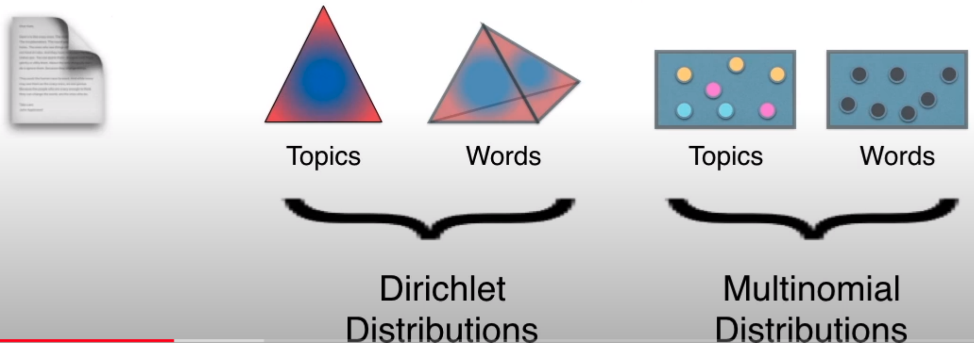
Dirichlet Distributions

$$f(x_1, \ldots, x_K; \alpha_1, \ldots, \alpha_K) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$



Probability of a document

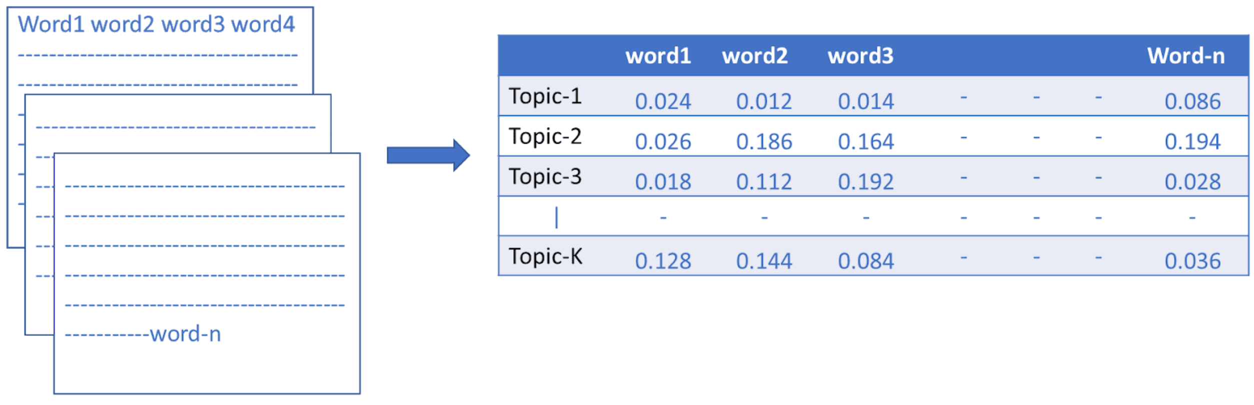
$$P(\boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\theta}, \boldsymbol{\varphi}; \alpha, \beta) = \prod_{j=1}^{M} P(\theta_j; \alpha) \prod_{i=1}^{K} P(\varphi_i; \beta) \prod_{t=1}^{N} P(Z_{j,t} \mid \theta_j)\, P(W_{j,t} \mid \varphi_{Z_{j,t}})$$



Working of LDA:

LDA assumes that a statistical generation process is created for each document. Each document is formed by a combination of topics and each topic is formed by a combination of words. At first, LDA assigns topics to each word completely at random. The topic assignment is done through Gibbs sampling after random assignment and this process is taken out till n iterations for improvement.

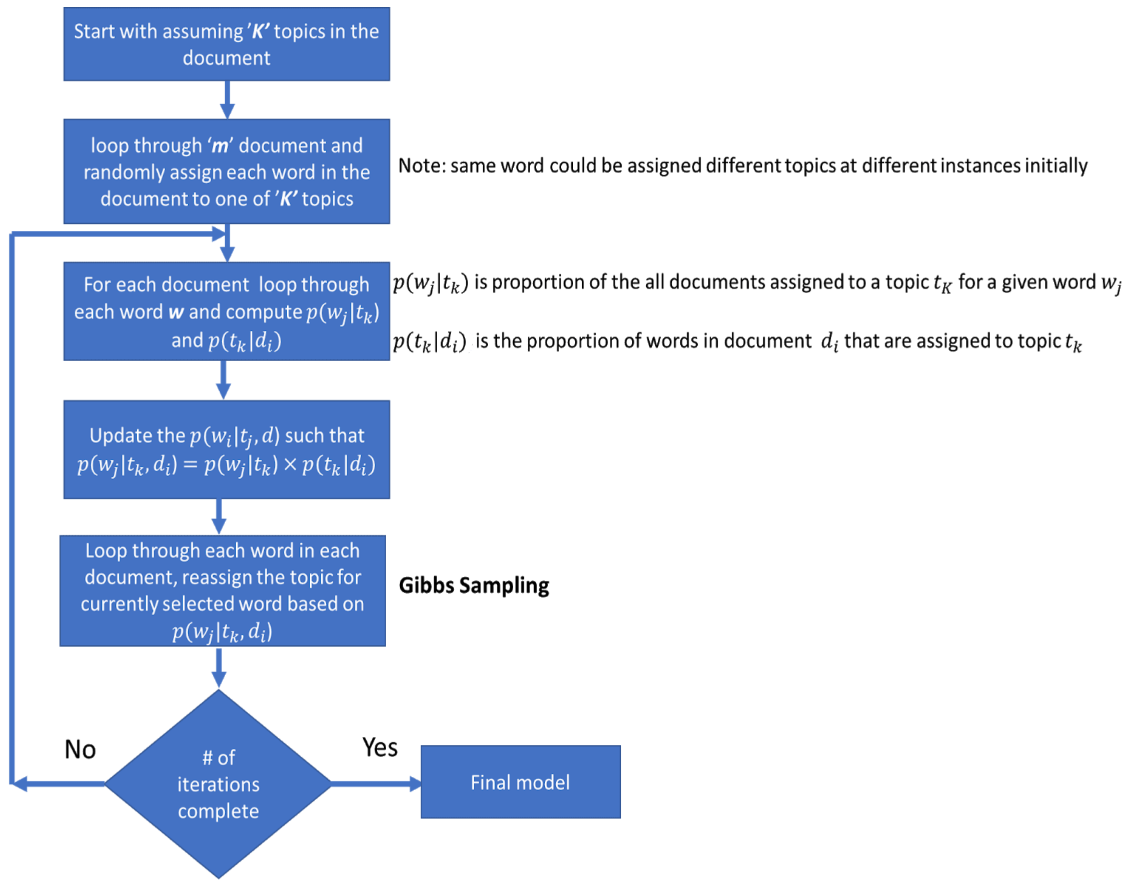
The task is based on the probability assessment as shown in the following figure The value of a word wj belonging to the topic tk is shown on the figure in each cell. The word and the topic indices are 'j' and 'k.' LDA does not recognize the order in which words and syntax information take place. It's just like a bag of words or a collection of words to handle documents.



	word1	word2	word3				Word-n
Topic-1	0.024	0.012	0.014	-	-	-	0.086
Topic-2	0.026	0.186	0.164	-	-	-	0.194
Topic-3	0.018	0.112	0.192	-	-	-	0.028
|	-	-	-	-	-	-	-
Topic-K	0.128	0.144	0.084	-	-	-	0.036

‘m’ documents with ‘n’ words

After the probabilities are estimated (we will see how these are shortly estimated), you can find the word collection that constitutes a given theme either by choosing top 'r' probabilities of words or by setting a probability threshold and only selecting words whose probabilities are higher than or equal to the threshold.


Start with assuming '*K*' topics in the document
loop through '*m*' document and randomly assign each word in the document to one of '*K*' topics
Note: same word could be assigned different topics at different instances initially
For each document loop through each word ***w*** and compute $p(w_j|t_k)$ and $p(t_k|d_i)$
$p(w_j|t_k)$ is proportion of the all documents assigned to a topic t_K for a given word w_j
$p(t_k|d_i)$ is the proportion of words in document d_i that are assigned to topic t_k
Update the $p(w_i|t_j, d)$ such that $p(w_j|t_k, d_i) = p(w_j|t_k) \times p(t_k|d_i)$
Loop through each word in each document, reassign the topic for currently selected word based on $p(w_j|t_k, d_i)$
Gibbs Sampling
No
# of iterations complete
Yes
Final model

Step-10 Resampling/ Reassign

For a given word w_j in a document d_i, find the topic '*k*' for which $p(w_j|t_k, d_i)$ is maximum and reassign the word to the 'k^{th}' topic

LDA has three hyperparameters;
1) The document-density topic's factor is 'α'.
2) 'β' factor of the topic density.
3) the number of topics 'K' to be taken into account.

The 'α' hyperparameter monitors the topics in the paper. The low value of 'a' means that fewer topics are expected in the mix and a higher value means that a higher number of the topics are expected in the mix.
The hyperparameter 'β' controls word distribution by topic. At lower 'β' values, the topics will probably have fewer words and more words at higher values.

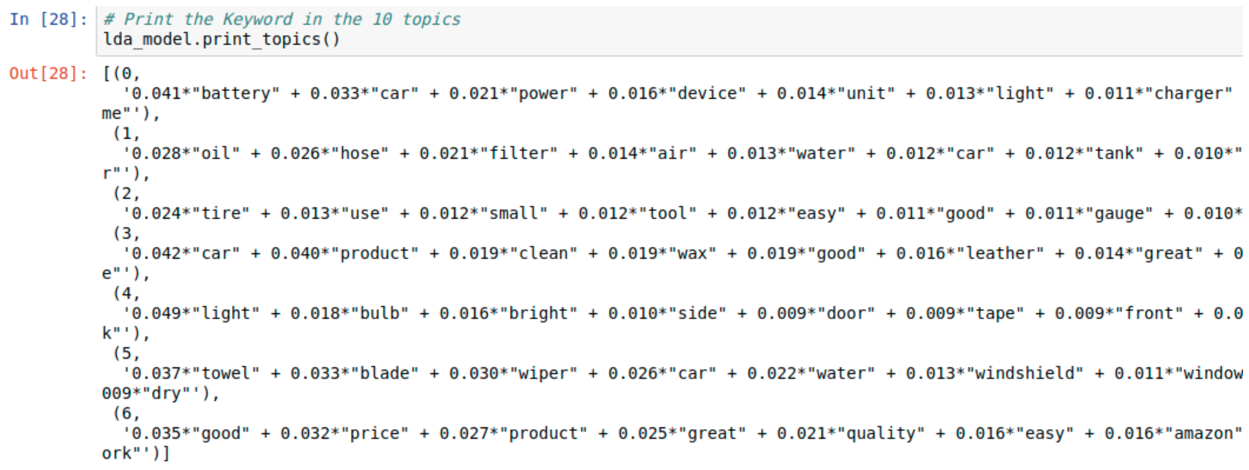
Chapter-3
Results & Discussion

3.1 Results & Discussion

We have successfully extracted Topics from the Amazon reviews dataset. Topic Modelling can be deployed on any customer service, online shopping, or any other application which includes reviews of customer or user. Through topic modeling, we are able to get better insights into what the customer is looking at, and also we can improve in areas where the product is inadequate.

The Amazon Automotive Dataset contains several reviews of persons who purchased car spare parts or anything related to the car. It is difficult to find out in which areas we need to improve to achieve maximum sales and customer satisfaction.

We have done all necessary preprocessing steps as discussed above, Later we have used LDA(Latent Dirichlet Allocation) for topic modeling in order to find the non-correlated topics. We can see the following 7 Topics in the following figures which have a probability of each word that is relevant to form each individual Topic.

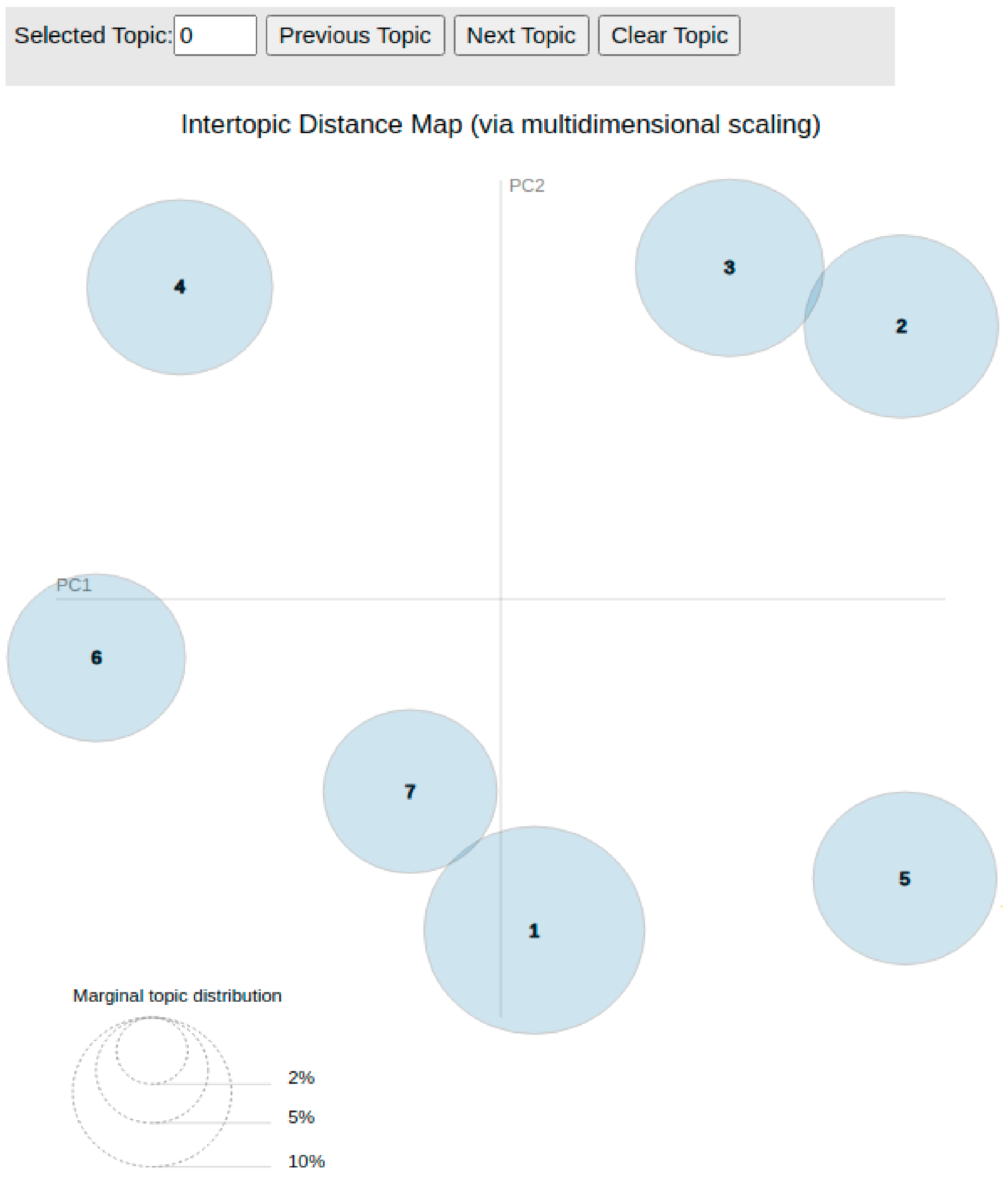


```
In [28]: # Print the Keyword in the 10 topics
         lda_model.print_topics()

Out[28]: [(0,
           '0.041*"battery" + 0.033*"car" + 0.021*"power" + 0.016*"device" + 0.014*"unit" + 0.013*"light" + 0.011*"charger"
         me"'),
          (1,
           '0.028*"oil" + 0.026*"hose" + 0.021*"filter" + 0.014*"air" + 0.013*"water" + 0.012*"car" + 0.012*"tank" + 0.010*"
         r"'),
          (2,
           '0.024*"tire" + 0.013*"use" + 0.012*"small" + 0.012*"tool" + 0.012*"easy" + 0.011*"good" + 0.011*"gauge" + 0.010*
          (3,
           '0.042*"car" + 0.040*"product" + 0.019*"clean" + 0.019*"wax" + 0.019*"good" + 0.016*"leather" + 0.014*"great" + 0
         e"'),
          (4,
           '0.049*"light" + 0.018*"bulb" + 0.016*"bright" + 0.010*"side" + 0.009*"door" + 0.009*"tape" + 0.009*"front" + 0.0
         k"'),
          (5,
           '0.037*"towel" + 0.033*"blade" + 0.030*"wiper" + 0.026*"car" + 0.022*"water" + 0.013*"windshield" + 0.011*"window
         009*"dry"'),
          (6,
           '0.035*"good" + 0.032*"price" + 0.027*"product" + 0.025*"great" + 0.021*"quality" + 0.016*"easy" + 0.016*"amazon"
         ork"')]
```

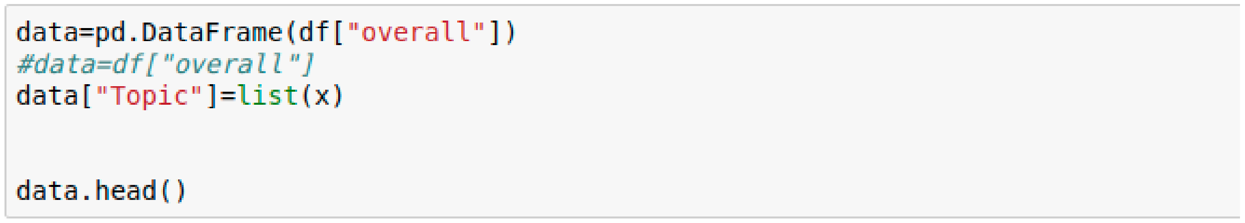
```
In [27]: import pyLDAvis.gensim
         pyLDAvis.enable_notebook()
         vis = pyLDAvis.gensim.prepare(lda_model, doc_term_matrix, dictionary)
         vis
```

Out[27]:



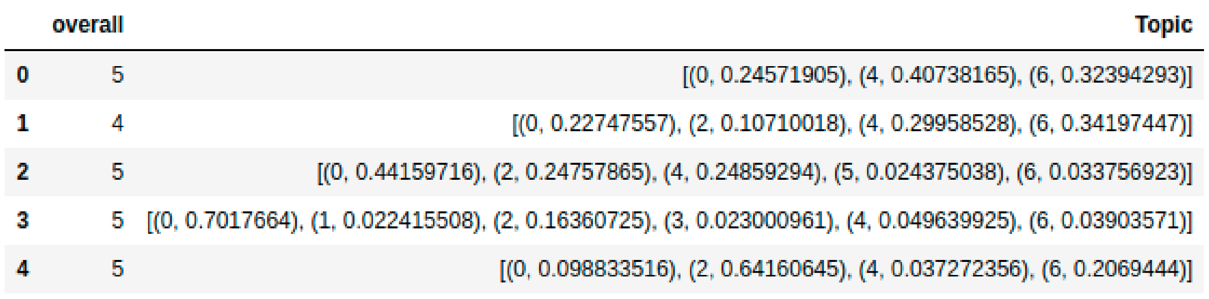
The above figure shows the visualization of different topics with their location in the vector space. We have found these 7 topics that don't overlap each other, Which means there are 7 topics that customers are interested in looking at. It gives us the straight results that customers are looking into these seven topics commonly. If we focus on these 7 topics we will be able to improve sales as well as customer satisfaction.

After topic modeling and finding relevant topics we have also found sentiments of each topic with their percentage. In order to find the sentiments, We have separated The overall rating along with the topics for which we don't need the probabilities. We can see the detailed explanation in the following output file.



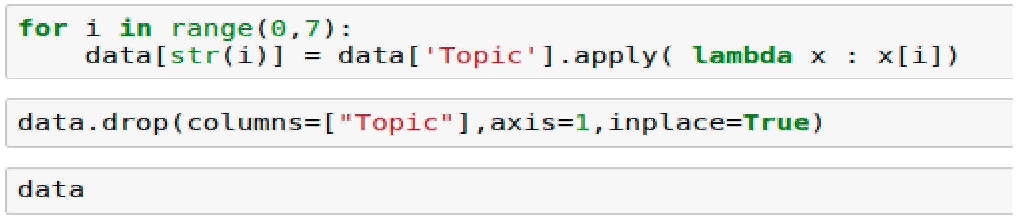
```
data=pd.DataFrame(df["overall"])
#data=df["overall"]
data["Topic"]=list(x)


data.head()
```



	overall	Topic
0	5	[(0, 0.24571905), (4, 0.40738165), (6, 0.32394293)]
1	4	[(0, 0.22747557), (2, 0.10710018), (4, 0.29958528), (6, 0.34197447)]
2	5	[(0, 0.44159716), (2, 0.24757865), (4, 0.24859294), (5, 0.024375038), (6, 0.033756923)]
3	5	[(0, 0.7017664), (1, 0.022415508), (2, 0.16360725), (3, 0.023000961), (4, 0.049639925), (6, 0.03903571)]
4	5	[(0, 0.098833516), (2, 0.64160645), (4, 0.037272356), (6, 0.2069444)]

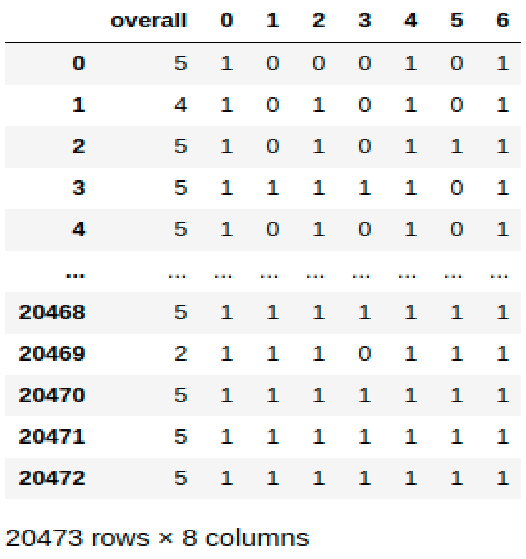
After separating the DataFrame we have done One Hot Encoding of all the topics in particular to each review index and their relevant topics along with the overall rating. We can see the following output file for better visualization.



```
for i in range(0,7):
    data[str(i)] = data['Topic'].apply( lambda x : x[i])
```

```
data.drop(columns=["Topic"],axis=1,inplace=True)
```

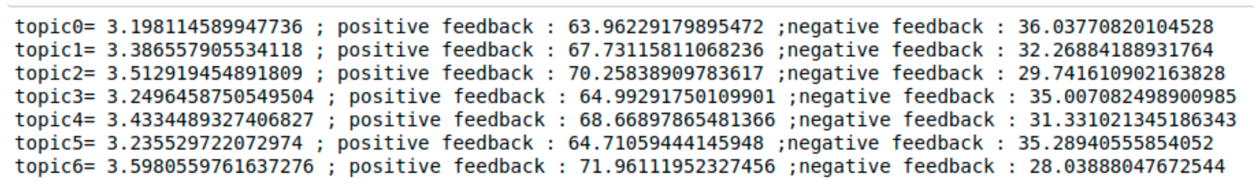
```
data
```



	overall	0	1	2	3	4	5	6
0	5	1	0	0	0	1	0	1
1	4	1	0	1	0	1	0	1
2	5	1	0	1	0	1	1	1
3	5	1	1	1	1	1	0	1
4	5	1	0	1	0	1	0	1
...	...	...	...	...	...	...	...	...
20468	5	1	1	1	1	1	1	1
20469	2	1	1	1	0	1	1	1
20470	5	1	1	1	1	1	1	1
20471	5	1	1	1	1	1	1	1
20472	5	1	1	1	1	1	1	1

20473 rows × 8 columns

We have calculated the average value of each topic along with the sentiment feedback which tells positive or negative sentiment and their respective customer percentage who have reviewed. We can see the details in the following output file.



```
topic0= 3.198114589947736 ; positive feedback : 63.96229179895472 ;negative feedback : 36.03770820104528
topic1= 3.386557905534118 ; positive feedback : 67.73115811068236 ;negative feedback : 32.26884188931764
topic2= 3.512919454891809 ; positive feedback : 70.25838909783617 ;negative feedback : 29.741610902163828
topic3= 3.2496458750549504 ; positive feedback : 64.99291750109901 ;negative feedback : 35.007082498900985
topic4= 3.4334489327406827 ; positive feedback : 68.66897865481366 ;negative feedback : 31.331021345186343
topic5= 3.235529722072974 ; positive feedback : 64.71059444145948 ;negative feedback : 35.28940555854052
topic6= 3.5980559761637276 ; positive feedback : 71.96111952327456 ;negative feedback : 28.03888047672544
```

At last, We have also named each topic according to their relevant words when visualized with their probabilities and occurrences. The names of the topics were

Topic 0: Electricals
Topic 1: Engine
Topic 2: Tires
Topic 3: Washing
Topic 4: lights
Topic 5: Car Glass
Topic 6: Value for Money

Therefore in the Amazon reviews dataset customer really needs to improve in the areas Electricals, Washing and Windshield, and glass which has many negative sentiments on these topics. Customers are much satisfied with the topics lights and value for money. Through this analysis, customers can make easy decisions, and also sellers can easily make analyses and improve in those areas.

Chapter-4
Conclusion & Future Scope

4.1 Conclusion

Topic Modelling on Amazon Reviews dataset has been successfully completed and extracted 7 different topics namely Electricals, Engine, Tires, Washing, lights,Car Glass, Value for Money. We need to focus on Electricals, Washing, and Car Glasses which has significant disappointment in the user reviews and has more negative sentiments. Customers are much satisfied in the areas of Lights and Value for Money which have good positive sentiment compared to other topics.

These results were achieved using the LDA(Latent Dirichlet Allocation) algorithm along with some important natural language preprocessing techniques. We have also successfully extracted each topic's sentiment along with the user's percentage by performing one hot encoding technique. This Topic Modeling can be deployed on an end-to-end platform such as customer reviews websites, food ordering websites, or any other application which involves customer reviews and feedback. Through this topic modeling, we can improve sales and ensure maximum satisfaction of customers. Through sentiments on each topic, we can make decisions using these sentiments whether to use or buy a particular service or product respectively.

4.2 Future Scope

Topic Modeling on Amazon Reviews can be improved in many areas. Following are the points which briefly explain which areas we can improve.

- We can make Topic Modeling a pipeline so that we can use it in any of the applications with required data.
- We can add a user interface to it so that users or customers can easily make decisions on the data.
- We can use Machine Learning for extracting Topics rather than using LDA

- We can use sentimental analysis on the range of 5 scales in order to get better insight rather than giving positive and negative sentiments.

References

1. Mohan, Vijayarani. Text Mining: Open Source Tokenization Tools: An Analysis. 2016.
2. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. 2003.
3. Tong, Zhou & Zhang, Haiyi. A Text Mining Research Based on LDA Topic Modelling. Computer Science & Information Technology. 2016.
4. Skorkovská, Lucie. Application of Lemmatization and Summarization Methods in Topic Identification Module for Large Scale Language Modeling Data Filtering. 2012.
5. Zhang, Yin & Jin, Rong & Zhou, Zhi-Hua. Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics. 2010.
6. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient estimation of word representations in vector space. 2013.