Amazon User Review Analysis with UI

Shiva Baghel[#1], Sonal Priya[#2], Harsh Raj[#3], Nirpesh Joshi[#4], Nishant Singh[#5]

[#]*B.Voc (Robotics &AI), Dayalbagh Educational Institute,*
Agra (282005), U.P, India

Notebook Link - https://github.com/Internship-BVoc/Amazon-User-Review-Analysis-using-UI

Abstract

Reviews are the most viewed section in-order to grab the information about a product like the material, efficiency etc. and reviews are more important for Amazon (MNC). In this paper, we classify reviews into categories by using several NLP libraries and algorithms also, extracted companies mentioned. As an addition, we have done sentiment and created a webpage for the ease of understanding. The acquaintance was provided to every customer about the products by this research.

I. Introduction

In today's World online shopping is a need for everyone. Amazon is one of the biggest names among the largest online vendors in the World. Reviews plays an important role while selecting the product. Till date amazon's reviews are only classified on rating (1-5) and the search bar prefers the product name not the review or the category to search a product. So, in this paper we explore some ways which can be used to improve the searching efficiency and ease to understand the review. By performing keyword extraction and Sentiment Analysis.

To do so, we have taken an Amazon review dataset contains about 40,000 reviews and 16 columns containing relevant information about the product. The dataset of related to toys and building material type products. The dataset was messy and reviews are not given as expected, we had extracted the reviews along with their rating and perform several Natural Language Processing (NLP) techniques to clean the reviews like remove punctuation/stop-words, lemmatization, html tags removal.

We worked on 3 model to get the best accuracy. Firstly, we tried Doc2Vec with Logistic Regression. Doc2Vec converts a review into vector so that it is easier to compare with other reviews. Secondly, TF-IDF a very famous approach in NLP. We applied TF-IDF on every review and find their cosine similarity based on the vectors of TF-IDF. Finally, we used Word2Vec model of Google and trained our own model on top of it.

At last we created a webpage to show the results in a better way, so that everyone can see our work in an interactive and friendly manner.

II. Related Work

The first category prediction was done on E-mails, we talk broadly than spam or not spam. "given the categories and timestamps of user's past emails, we want to predict the probability distribution over the categories of emails that the user will receive in the next n days." written in paper.

Mr. Ashish Kumar did a work on category prediction of clothes, and he took a vast dataset of clothes name with their categories and applied several pre-processing. He trained his model on Tf-idf and get a accuracy of 78%, he said "It is a painstaking adventure to build this categorization model but looking at the value it has added to our business, it seems totally worth it. With these new ML-powered features, we are confident of adding a lot of value to customer's online shopping experience with our smart shopping options."

Mr. Amadeus Magrabi did a great work he categories wearables by their photos using a perfect dataset of category and photos, he used several computer vision techniques. He also uses tf-idf and got an accuracy of 90% on the trained model, but if the tries new samples than the accuracy fall down to 70% - 80%.

III. Literature Survey

MNCs like Amazon must have considered every aspect of user's desire. If we talk about the searching algorithm and reviews analysis, then the lack of categories-based search and review classification is observed. So, we have researched on this topic that how to can predict the category given a review and search products based on that category.

Till now in amazon review or search bar does not support category-based search. We took a dataset of amazon (basically of toys) and tried various Machine Learning techniques to do so, here is our research-

A. Data –

We took a dataset of amazon based on kids' toys. The dataset is of Amazon with about 10,000 rows of different toy companies and 16 columns with relevant information.

The column contains information

[*product_name, manufacturer, price, number_of_reviews, average_review_rating, amazon_category_and_sub_category, customer_reviews, sellers, 'uniq_id', 'number_available_in_stock', 'number_of_answered_questions', 'customers_who_bought_this_item_also_bought', 'description', 'product_information', 'product_description', 'items_customers_buy_after_viewing_this_item', 'customer_questions_and_answers'*]

We have in a total –

- 10,000 toy products
- 256 unique categories are present
- 40,000 reviews are extracted from the 10,000 rows
- 2645 unique manufacturer makes 10,000 products

B. Dataset Insights and feature extraction (pre-processing)-

- Most of the product are of category "Hobbies > Model Trains & Railway Sets > Rail Vehicles > Trains"
- Hornby makes highest 171 products.
- "DJI Phantom 2 with H3-3D Gimbal" has the highest price 995.1 euro.
- Positive Biased

Firstly, for filtering and future processing we identify the unnecessary columns, the columns which do not contain any relevant information like reviews, rating etc. and are burden for future steps. So, we dropped the columns listed below –

```
['uniq_id', 'number_available_in_stock', 'number_of_answered_questions', 'customers_who_bough
t_this_item_also_bought', 'description', 'product_information', 'product_description', 'items
_customers_buy_after_viewing_this_item', 'customer_questions_and_answers']
```

After that, we extract the review from the "reviews" columns. The reviews were present in below order with date, rating, and name of the reviewer.

E.g.
"on 6 April 2014 // Part of the magic for me growing up as a boy was to buy (or be given) the new Hornby catalogue every year, even if it included 90% of the same products as the previous year. I've still

got my old ones dating back to the 70s and 80s somewhere. These days the catalogue is especially informative in that it tells you the vintage of the rolling stock which is useful if you are dedicating your railway to one particular era and train company. | Amazing detail fabulous photography. // 5.0 // 11 April 2015 // By Richard"

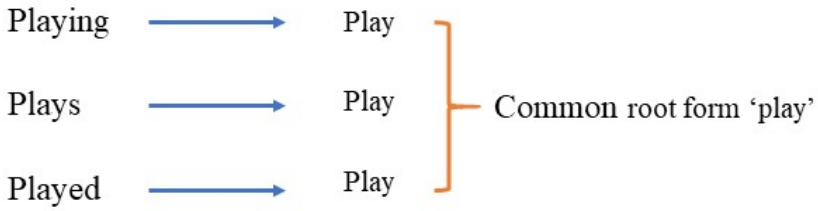
The following steps are done to get the reviews and rating –

- Split the data by "//".
- Find rating and reviews in the list (generated by above step)

We get about 40,000 reviews after splitting the column, this is the reason why you must understand your data before doing any pre-processing.
Now, we apply some feature engineering to prepare our data for training.

Pre-processing –

- Tokenization - Word tokenization is the process of splitting a large sample of text into words. This is a requirement in natural language processing tasks where each word needs to be captured and subjected to further analysis like classifying and counting them for a particular sentiment.
- Punctuation Removal (./?@%#$|) – Punctuation are the marks, such as full stop, comma, and brackets, used in writing to separate sentences and their elements and to clarify meaning. Deleting punctuation reduces the ability of the follow-on semantic parsing functionality.
- Stop Words Removal (and, or, hi, at) - Stop words are often removed from the text before training deep learning and machine learning models since stop words occur in abundance, hence providing little to no unique information that can be used for classification or clustering.
- Lemmatization



[1]

- Html tags Removal – (https://saja, www.dfjdjf.in) are done using several Natural Language

Processing library present in library –

a. NLTK
b. Spacy

After pre-processing we got our processed data and we are ready to perform some NLP and ML techniques.

C. Research Overview –

Before dive into visualization and methodology, lets understand whole research overview with the help of a flow chart.
The given flowchart shows each step we had done in-order to predict the category and relevant entities.

- Data cleansing – pre-processing of data and feature extraction
- Visualization – understand relationship between columns
- Model training – or methodology –

  Training of model using final dataset without any impurities or outliers.
  Approaches used –
  - Doc2Vec with Logistic regression
  - TF-IDF with Cosine Similarity
  - Word2Vec with personalized code

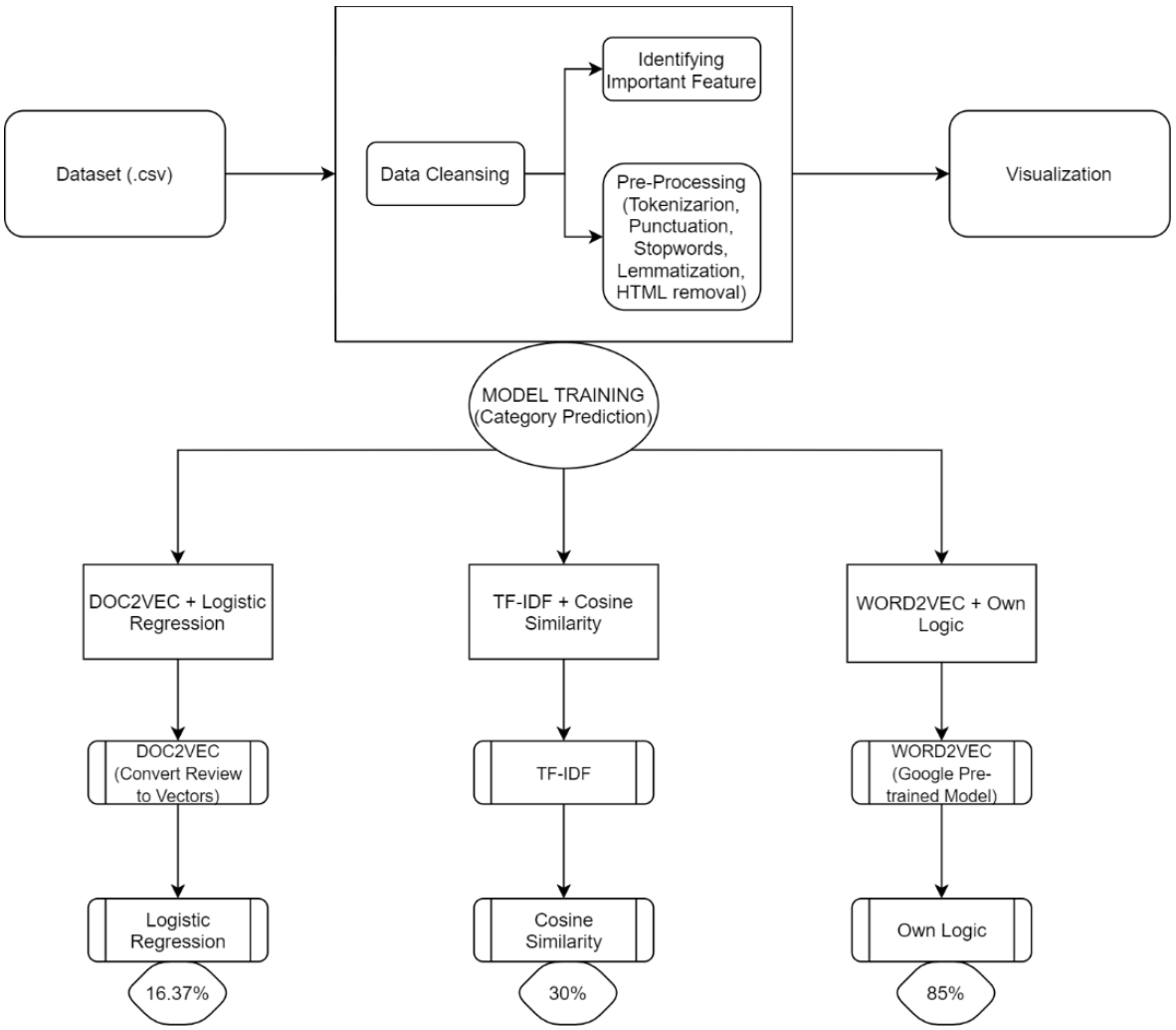


Fig. 1 Process Summary

D. Visualization-

Before training we did some visualization on our data to understand the trend and relationship between columns. Visualization is performed by some python library –

i. Matplotlib

ii. Seaborn

iii. Altair

iv. Plotly

a. Word Cloud (Positive and Negative words) w.r.t manufacturer-

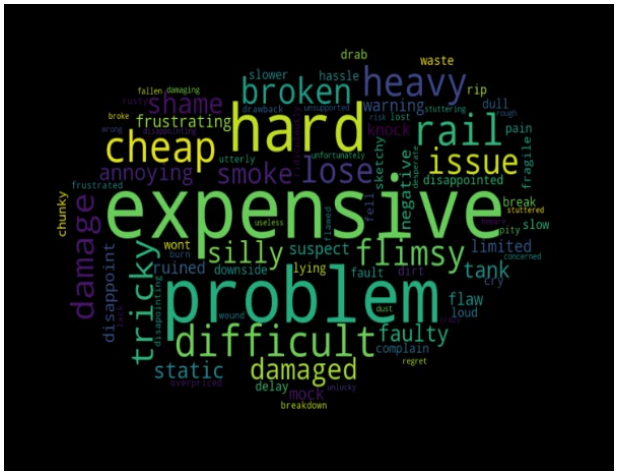


Fig. 2 Postive Words **Fig. 3 Negative Words**

We used reviews of Hornby (manufacturer) to generate these word clouds.

- From Fig.2, words like “better” and “work”, are used most, that means the toys are in perfect condition and ready to use.
- From Fig.3 “expensive” is most used, means that most of the products are costly.
- Also, from Fig.3 we get that there is some problem with products as words “hard”, “problem”, “difficult”, “flimsy” are more visible.

b. Manufacture v/s price based on category –

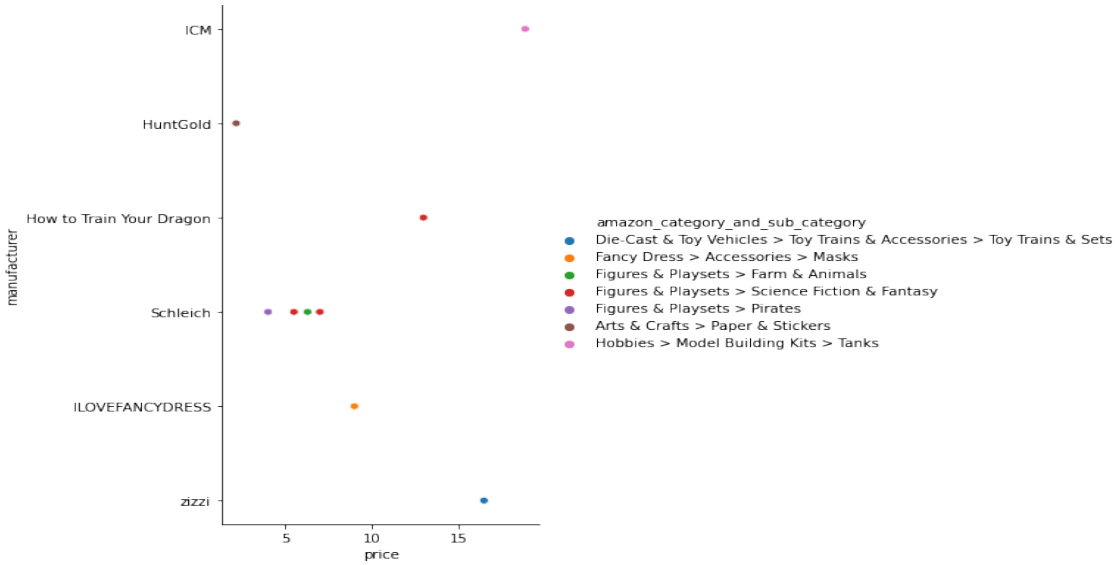


Fig. 4 Top manufacturers

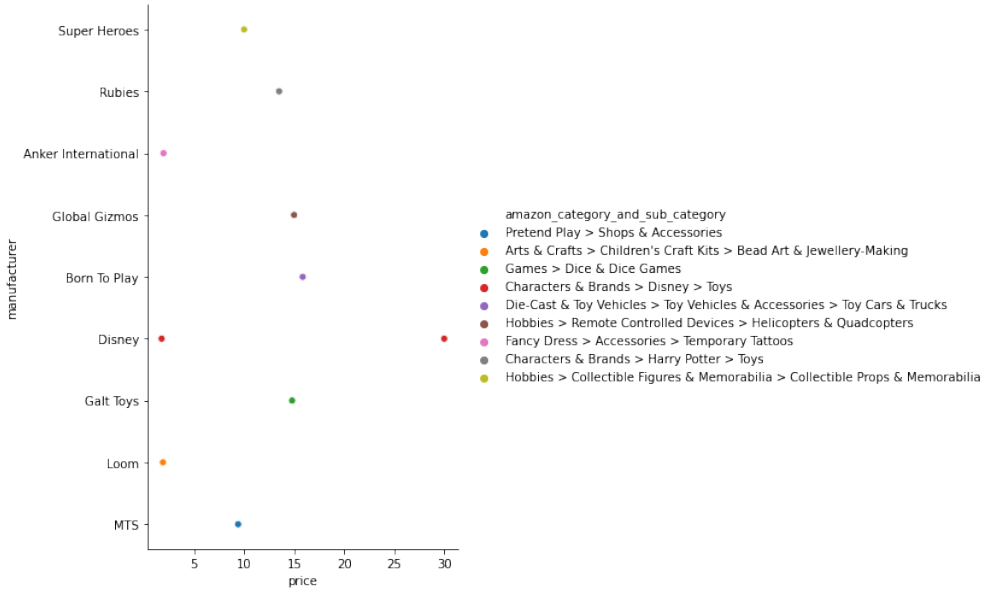


Fig. 5 Least Manufacturers

From the above graph we got to know that-

- Some manufacturer sales more than category of products
- Price difference can be seen b/w products of same category and manufacturer
- Categories like art & craft, hobbies are same in top rated and low rated reviews

c. Manufacturer v/s Rating-

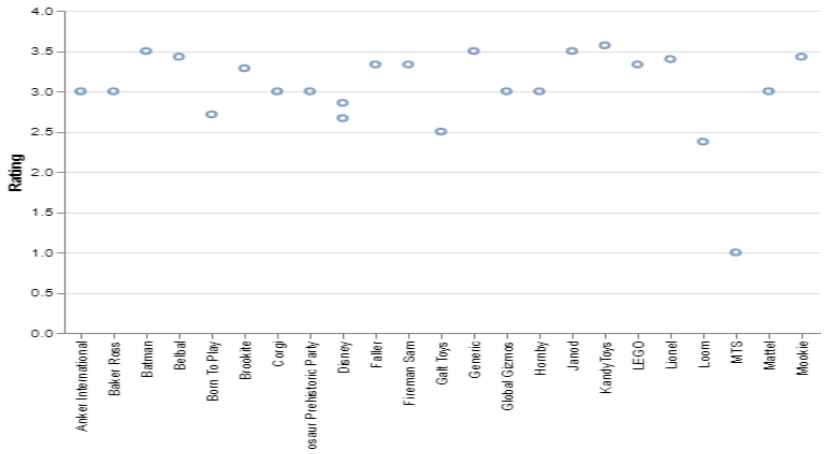


Fig. 6 Ratio for middle manufacturer

- This graph shows (24 manufacturer) the review with rating below 3.5, from 20,000 reviews only these much are negative or low rated. So, the dataset is positive biased.
- We got that most of the reviews are rated above 3.5.
- To solve that issue we take a fair ratio of reviews for sentiment and category prediction.

d. Price v/s Rating-

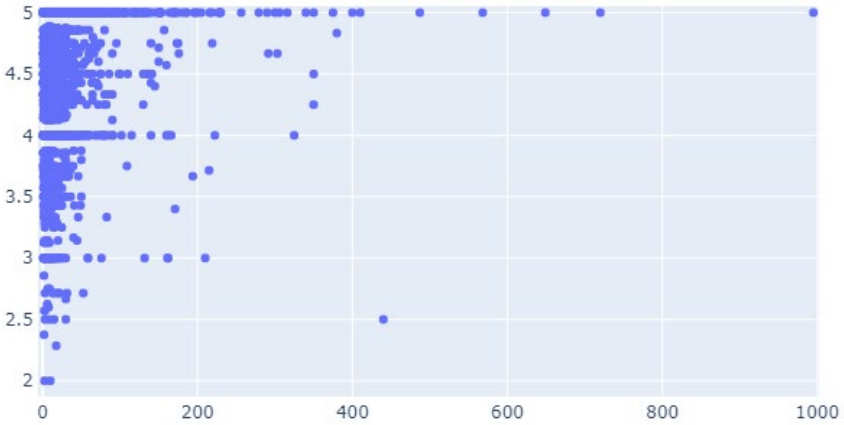


Fig. 7 Price v/s Rating Ratio

- From Fig. 7, It is clearly observed that people buy cheap products with variable rating between 2.5-4.5.
- Most selling products are having price below 100 euro.
- The product with higher price is rated as 5, that means the quality is Good.

e. Most frequent words (Bag of Words)-

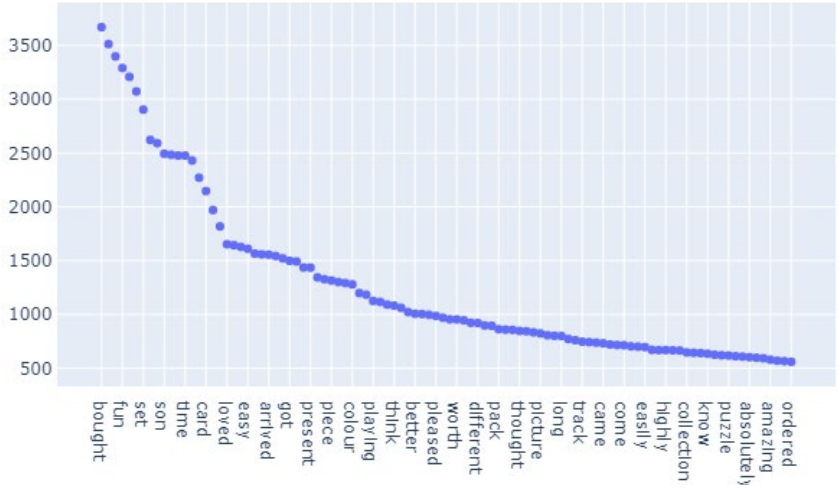


Fig. 8 Most Frequent Words

- Fig. 8 is plotted for top 100 Most occurring words.
- The word “bought”, “fun” are seen most frequent among all the reviews.
- It is clear from the word “son” that most of the products are bought by parents for their kids.

f. Rating v/s category based on price-

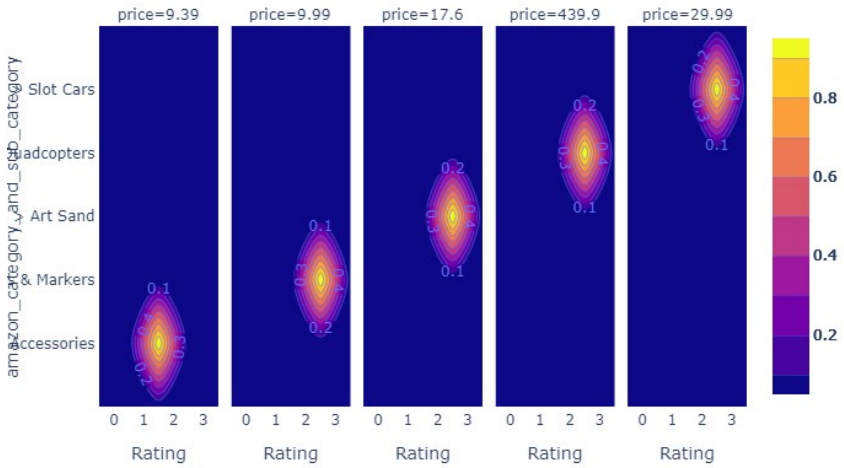


Fig. 9 Density Graph showing impact of price on rating

- This density graph shows that the category with higher price and count has higher rating.
- “Cars” and “quadcopters” have highest rating.

E. Entity Detection:

Entity extraction, also known as entity name extraction or named entity recognition, is an information extraction technique that refers to the process of identifying and classifying key elements from text into pre-defined categories.[8]

We used entity extraction to get the company names present in our reviews. There are various approaches and libraries available in python.

i. NLTK – using chunking we can find the entity.

Fig. 10 NLTK Output [2]

ii. Spacy – displacy will help to find the entity.

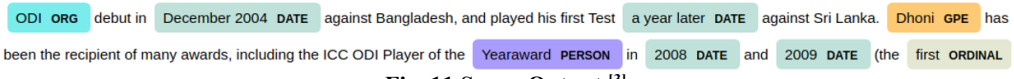


Fig. 11 Spacy Output [3]

iii. Polyglot –

```
I-ORG [u'Israeli']
I-PER [u'Benjamin', u'Netanyahu']
I-LOC [u'Iran']
```

Fig. 12 Polyglot Output

- The ORG., PER, LOC, Value are extracted from the given sentences
- We need only ORG. because this contains the Company or ORG name
- We used a hybrid approach, smartly combining all the result from these 3 libraries.

IV. METHODOLOGY

After completing the pre-processing and finding trend by visualization we have some image clear in our mind that how to approach the classification. To classify a review into its sub-categories we took 3 approaches that had have mentioned above. Now we see each approach one by one.

1. Doc2Vec with logistic regression –

   Doc2Vec –

   the goal of doc2vec is to create a numeric representation of a document, regardless of its length.

   This algorithm is used to convert document or sentences into vectors and finding similarity between sentences or document. But using only Doc2Vec is a very bad decision.

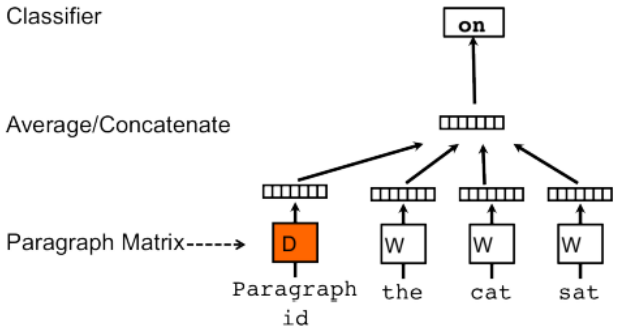


Fig. 13 Doc2Vec Example [4]

- Fig. 13 demonstrate that para. Id helps while predicting the perfect word to be fitted in the given set of tokenized words.

Accuracy – 14% (approx.)

So, we searched and found that by using Machine Learning algorithms the accuracy can be increased, logistic regression gives the best result-

Multinomial Logistic regression –

- Three or more categories without ordering.
- Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist.

	precision	recall	f1-score	support
Arts & Crafts > Art & Craft Supplies > Glitter	0.50	1.00	0.67	1
Arts & Crafts > Art Sand	0.10	0.18	0.13	45
Arts & Crafts > Blackboards	0.00	0.00	0.00	8
Arts & Crafts > Children's Chalk	0.00	0.00	0.00	3
Arts & Crafts > Children's Craft Kits	0.05	0.03	0.04	32
accuracy			0.06	3664
macro avg	0.02	0.02	0.02	3664
weighted avg	0.05	0.06	0.05	3664

Fig. 14 LR result

- From Fig. 14, our LR model predicts only some of the category correctly.

Accuracy – 16.37% (approx.)

2. TF-IDF with Cosine Similarity –

   After getting unsatisfactory result from Doc2Vec approach, we tried this

   TF-IDF (Term Frequency - Inverse Document Frequent) – is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The importance increases

proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the TF-IDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query.

i. **TF: Term Frequency**, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization:

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

ii. **IDF: Inverse Document Frequency**, which measures how important a term is. we need to weigh down the frequent terms while scale up the rare ones, by computing the following:

IDF(t) = $\log_e$ (Total number of documents / Number of documents with term t in it). [5]

Cosine Similarity –

Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space.[13]

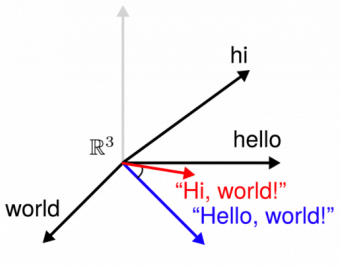


Fig. 15 [6] **Cosine Measure**

- From Fig. 15, the angle between "hi" and "hello" is less than "world", but if we see "Hi, World" and "Hello, World" the angle is very acute, because "world" are same in both the sentence so, only the angle between "hi", "hello" matters, which we can see are same.

This is a very popular approach while solving these types of problems, but in our case, we have a huge number of categories (256) so, many of the reviews of different category seems similar. We can see the result with the help of this heat map-

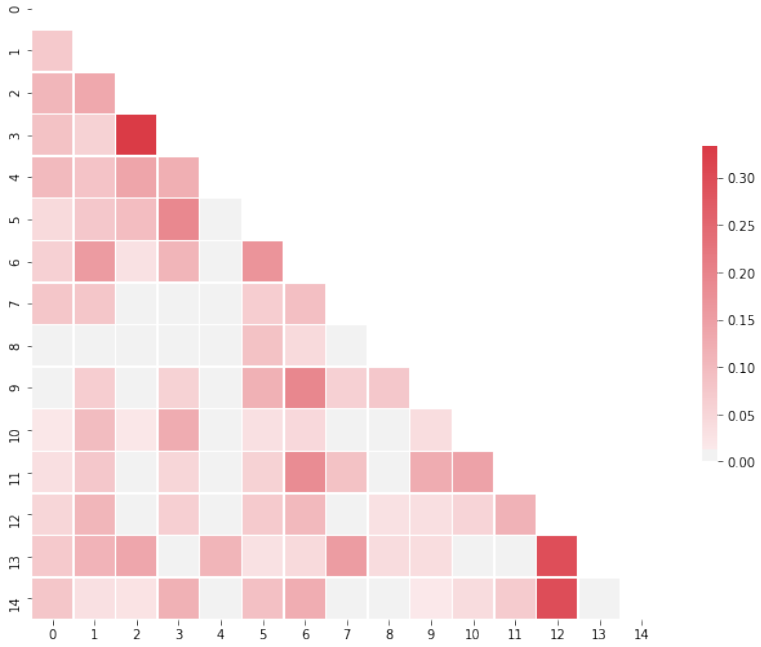


Fig. 16 Relation b/w 2 sentences

We can see that some of the reviews are corelated to each other, but the highest similarity score is **30%** which is penurious and perform very mediocrely. therefore, we chose to move on to some other approach.

3. Word2Vec with personalized code –

These 2 approaches do not seem very help full. After doing some research, we found a technique called Word2Vec and is very popular nowadays in industries.

Word2Vec – Word2vec is a two-layer neural net that processes text by "vectorizing" words. Its input is a text corpus and its output are a set of vectors: feature vectors that represent words in that corpus. While Word2vec is not a deep neural network, it turns text into a numerical form that deep neural networks can understand.[15]

Word2vec is an algorithm, present in Gensim library of python, is basically an algo. which converts word into vector and links similar type of words in a document. There is a pre-trained Word2Vec model by Google of about 3 million words in 300 dimensions. We trained our data on top this model in-order to get more accurate result.

```
king - man + woman = queen
```

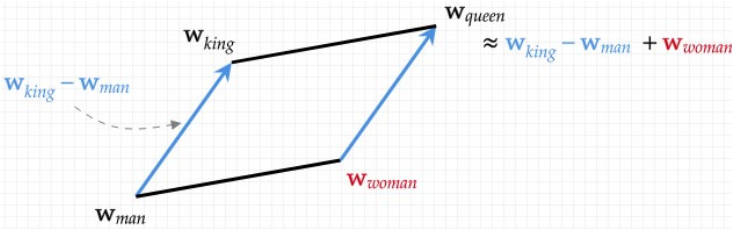


Fig. 17 Example Word2vec [7]

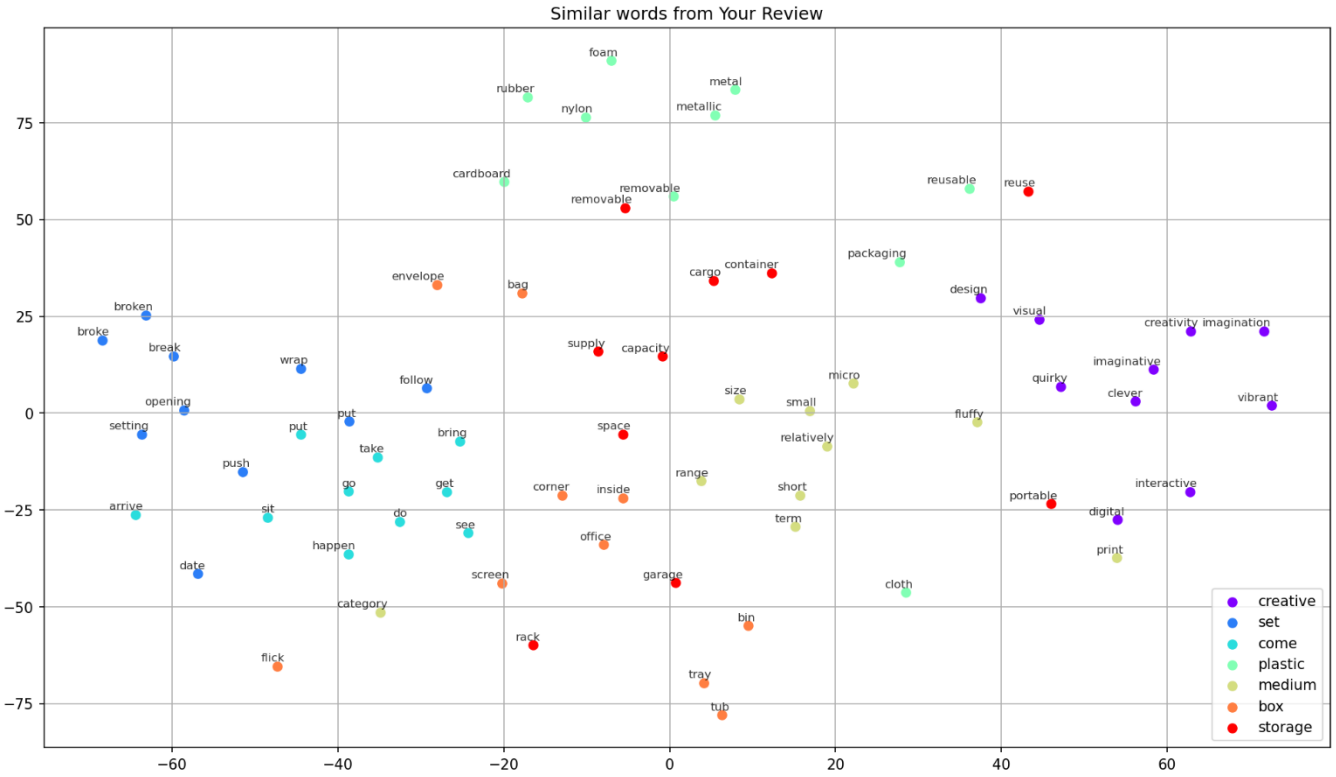


Fig. 18 TSNE graph

After the model training, we just saved the model, and move to write our own personalized classification code for given category and review. To do so we first extracted the company from the given review and matches the top words with their similar words in that review with the reviews of the categories related to the company and return the most relevant category for that review.

If the company was not given, then we categorize similar categories into a list and then we compare the top words of given review with the category reviews and returns most relevant category.

Overall Model Accuracy is 85% which is very effective, and Model performs best.

V. User Interface

For a better experience and understanding of our model, we decided to create a webpage using Flask (python library). It contains-

i. Text Area
ii. Buttons – each button for different task Category prediction, Company prediction, Keywords, Sentiment.

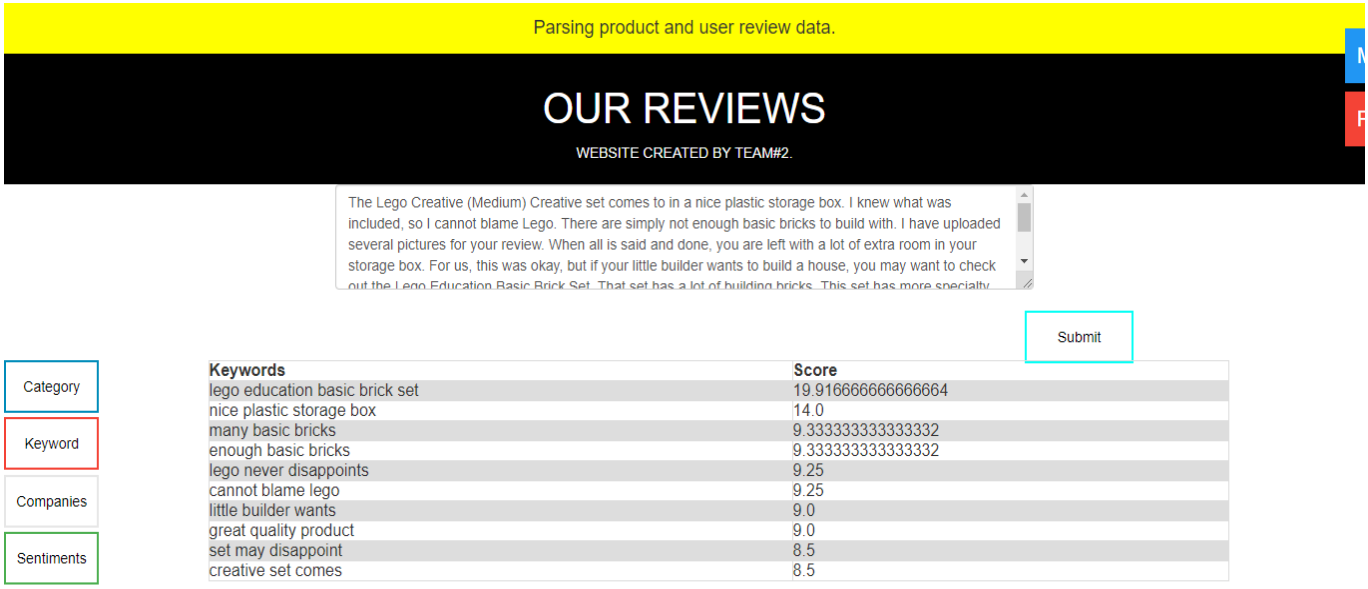


Fig. 19 Webpage

- We can see the beautiful webpage with heading of research, and button that gives relevant result.
- Also, after clicking the button we get a table showing the results with their relevant result

VI. Result

The purpose of this research is to provide customer pre-sorted reviews with correct extracted information. If we believe human psychology researches, then information given in points or in form of table is clearer than a paragraph. This is the main reason of classification of reviews and classification on the category is a tough job but once it is done it helps many of the customers to choose the product easily.

The result of our research is as follows-

Word2Vec performs best for classification job and gives very satisfying result in comparison with TD-IDF & Doc2Vec.

Our model summarizes the review into these tables of category, sentiment, keyword, and companies.

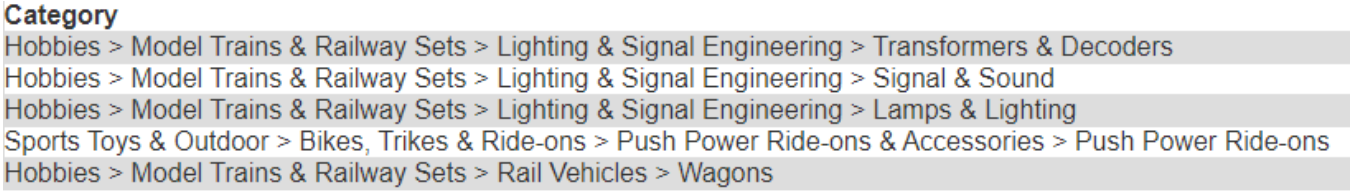


Fig. 20 Relevant Categories

- From the fig. we can see the predicted category by our word2vec model, these categories are arranged in the decreasing order of relevancy.

Sent/Words	Sentiment
Full Text	0.9439
lego education basic brick set	1.0😐
many basic bricks	1.0😐
lego never disappoints	0.522😊
little builder wants	1.0😐
set may disappoint	-0.574😢

Fig. 21 Sentiment Result

- Sentiment of full text entered and top 5 keywords
- These are the sentiment result using VADER, also for better understanding we added emojis.

VII. CONCLUSION

1. After all that, now comes the main part of extraction the useful information firstly we extracted entity from the given reviews. To do so, we used a hybrid approach like we merged the result coming from all the 3 techniques (NLTK, Spacy, Polyglot).

2. Category prediction, we have used 3 techniques to predict category –
   a. Doc2Vec with Logistic regression – this gives best result for Doc2vec using some Machine Learning Algorithm.

      Only Doc2Vec will give you accuracy of 14 %
      With logistic regression accuracy 16.37%

   b. TF-IDF with Cosine Similarity – TD-IDF converts word into vectors with its score of occurrences.
      Cosine Similarity – measures the angle between 2 words or sentences and gives back the values. Value between 0.9 – 1.0 is best for any word or sentence.

      Accuracy 30%

   c. Word2Vec with personalized code – Word2Vec is most used technique nowadays in NLP, this technique learns the similar words present in the dataset to a particular word and we can use them for understanding new sentences.

      Accuracy 85%

VIII. FUTURE SCOPE

Word2Vec has generated the best accuracy which is 85%. We completed this research project in 1 month. In future if we get the opportunity to work on this model again then, we try to implement some higher approaches for prediction like RNN with LSTM, and CNN (**Convolutional Neural Network**). Though both approaches guaranteed give more accuracy as the CNN and RNN both are trained on each and every relation between the input features. I think 90% - 95% can easily be reached by using these neural nets.

This model can be used in any search bar (Google, Bing etc.) with slight changes. Also, if someone wants then after some parameter changes, product and sellers of a particular product can also be predicted, but it needs RNN or CNN to do so.

IX. REFERENCES

[1] “Hafsa Jabeen” Stemming and Lemmatization in Python – DataCamp October 23, 2018
[2] How to Do Named Entity Recognition Python Tutorial
[3] Susan Li “Named Entity Recognition with Spacy”Aug 17, 2018
[4] Understanding document embeddings of Doc2Vec May 14, 2018
[5] K. Sparck Jones. "A statistical interpretation of term specificity and its application in retrieval". Journal of Documentation, 28 (1). 1972.
[5] G. Salton and Edward Fox and Wu Harry Wu. "Extended Boolean information retrieval". Communications of the ACM, 26 (11). 1983.
[5] G. Salton and M. J. McGill. "Introduction to modern information retrieval". 1983
[5] G. Salton and C. Buckley. "Term-weighting approaches in automatic text retrieval". Information Processing & Management, 24 (5). 1988.
[5] H. Wu and R. Luk and K. Wong and K. Kwok. "Interpreting TF-IDF term weights as making relevance decisions". ACM Transactions on Information Systems, 26 (3). 2008.
[6] Selva Prabhakaran “Cosine Similarity – Understanding the math and how it works (with python codes)”
[7] Carl Allen and Timothy Hospedales “King - man + woman = queen: the hidden algebraic structure of words” July 10, 2019