SENTIMENT ANALYSIS ON AMAZON PRODUCT REVIEWS

Venkatavaradan Raghuraman
CSE Dept.
PES University
Electronic City Campus
Bangalore,India
venkatavaradan.r@gmail.com

Sanskriti Pattanayak
CSE Dept.
PES University
Electronic City Campus
Bangalore,India
sanskriti.pattanayak@gmail.com

Avinash VK
CSE Dept.
PES University
Electronic City Campus
Bangalore,India
avinash2000vk@gmail.com

Akshay C Patil
CSE Dept.
PES University
Electronic City Campus
Bangalore,India
14.akshaycpatil@gmail.com

***Abstract*—This paper showcases a comparative study between different machine learning models to perform sentiment analysis on the customer reviews of Amazon products in the Electronics category. The primary models we will look into for our analysis are Support Vector Machines, Naive Bayes Classifier, Random Forest Classifier, BERT Model[7], Stochastic Gradient Descent (SVM Linear) and Linear SVC models.**

Keywords : Natural Language processing(NLP), Sentiment Analysis , Tokenization, Product reviews

I. INTRODUCTION

E-Commerce has emerged to become one of the biggest industries with a tremendous shift from offline shopping to online shopping. This trend is seeing an exponential rise and more and more people prefer online shopping for factors such as convenience and discounted

With new products popping into the market daily customers depend highly on customer reviews of products on eCommerce sites to make decisions about their purchase. It is important for brands to look into the kind of reviews products get and how these affect the way the product performs in the market.

These reviews take into account various aspects of the product namely the prize, the brand, the features, the competitive products, etc. We want to extract key performance indicators from the review data and extract useful reviews that will help the customers when shopping online.

II. DATASET

A. About the dataset

For this project, we decided to go with the 2018 updated version of the Amazon review dataset[1] released in 2014. It includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

B. Review Data

- The total number of reviews is 233.1 million (142.8 million in 2014).
- Current data includes reviews in the range May 1996 - Oct 2018.

III. LITERATURE REVIEW

A. Sentiment Analysis on Large Scale Amazon Product Reviews[2]

Tanjim Ul Haque & Nudrat Nawal Saber & Faisal Muhammed Shah , Dept of Computer Science and Engineering, Ahsanullah University of Science & Technology , Dhaka , Bangalore (2018)

The paper conducts analysis on Amazon Review dataset which makes it an ideal reference point. They have used Multinomial Naive Bayesian and SVM as the main classifiers for analyzing reviews. Using active learning (which is a semi-supervised learning algorithm) helps avoid bottleneck situations of unlabeled data as the model chooses what data it learns from. We find this paper very insightful for the fact that they have given a comparison between how well different approaches(SVM, MNB, Stochastic Gradient Descent, Random Forest etc.) faired for the particular dataset by comparing the accuracy, precision, recall, and F1 score.

B. How to Ask for a Favor: A Case Study on the Success of Altruistic Requests[3]

Tim Althoff , Cristian Danescu-Niculescu-Mizil , Dan Jurafsky Stanford University, Max Planck Institute SWS

The Paper was published in icwsm, and involves a very interesting concept of the Random Acts of Pizza, where people request pizza and a random stranger can buy them one. They have analyzed what factors contribute to a successful pizza request using beginner friendly models (L1

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

penalized Logistic Regression(Friedman, Hastie, and Tibshirani 2010)), visualizations, and Analytics. This paper helps students who are new to the world of research to ease in and get to know the general theme of research papers in general.

C. *Sentimental analysis of product reviews[4]*

Najma Sultana &Pintu Kumar & Monika Rani Patra & Sourabh Chandra and S.K. Safikul Alam (2019)

This research paper aims at running sentiment analysis on customer reviews and labeling text as either "Positive"," Negative" or "Neutral". The paper discusses a theoretical approach to sentimental analysis and analyses different algorithms for the same with their corresponding accuracies. It also gives a brief history of different other approaches to sentiment analysis techniques. The solution described in the paper involves constructing an ML model through three major parts: Data Filtration, Training model, and testing model. Data filtration involves pre-processing the text to remove unwanted items and using only relevant textual content for the model. In training, all the feature words(verbs, adverbs, and adjectives) are extracted and then classified. Training a dataset to classification algorithms like Naïve Bayes classification algorithm, Linear Model algorithm, SVM algorithm, and Decision tree is also done to compare accuracies. In the testing phase, the user input(review) is mapped to the saved feature set. Feature extraction is done using the frequency of occurrence.

D. Amazon Reviews Sentiment Analysis: A Reinforcement Learning Approach[5]

Roshan Pramod Samineedi Joseph(2020)

The researcher wanted to classify Amazon Feedback into binary class and likely into multi-school utilising reinforcement learning and pre-trained BERT model. The researcher utilises the product-based data collection from amazon.com. The study aims to evaluate and predict the sentiment behind the analysis using algorithms such as BERT,LSTM and to enhance learning, classifying it as positive, negative and neutral. LSTM(Long -short-Term Memory) is a specific form of RNN to avoid long-term dependency problem.It processes data passing on information as it propagates forward. The paper tries to come to a conclusion as to why this model is an apt approach for Sentiment Analysis.

IV. PREVIOUS WORK AND CHALLENGES FACED

Previous iterations of our work contains testing multiple methods of data cleaning and preparation in order to obtain the cleanest possible text data. In addition to the basic preprocessing and data cleaning we had done to the text data, lemmatizing, tokenization and removal of unclean data(tags,punctuation, stopwords etc.) were performed. This alone ended up increasing the accuracy by more than 18 points.

EDA remains the same for the majority.

The main changes between previous and current iterations of our work are changes to the hyperparameters of all models tested to improve accuracy even by a bit, inclusion of cross validation testing for all models and increase in dataset size.

Major problems that we ran into were only during the data collection part. Combated by a 20gb dataset with an 11gb Metadata dataset that did not run on any of our local machines led us to explore alternatives where we can do some basic cleaning and fragment the dataset into a more workable size. Furthermore, the dataset was originally in JSON format which did not help with the memory limitation problem we faced. We solved this problem by running a SageMaker notebook with 256gb ram to parse, merge, clean and fragment the dataset. Other than this there were no major roadblocks. Models and hyperparameter tuning went without too many hitches.

Fragmenting the data proposes a new problem of the sample not representing the population. We choose to use a simple random sample of the population for our analysis.

In order to obtain the true overall sentiment towards a product, we need to assume that all customers that buy the product leave a rating/review. Practically this is not true. The majority of customers that buy a product dont review or rate the product. The assumption would be that the majority of reviews tend to be on extremes of either side, but it turns out that follow up emails and marketing for a product net multiple reviews in the range of average (3 stars) to excellent (5 stars). On the other hand, the length of a text review tends to be larger when the person is writing a piece criticizing the product. This is clearly seen in our EDA, and the dataset itself tends to hold some bias towards positive reviews.

V. DATA-PREPROCESSING

TABLE I. PRODUCT REVIEW DATASET

Data attributes			
Attribute Name	***Data Type***	***Attribute Name***	***Data Type***
overall	float64	reviewer Name	object
verified	bool	reviewText	object
reviewTime	object	summary	object
reviewerID	object	unixRevie wTime	int
asin	object	vote	object
style	object	image	object

Fig. 1. Data attributes and corresponding data types.

A. Data Cleaning

- The first step is to look into duplicates, missing value, and inconsistent data.
- We predict the importance of columns and drop/keep them accordingly, mainly retaining only text data.
- The number of rows containing NANs were extremely insignificant to the total number rows we had, so we dropped them.
- Renaming the columns, and reassigning values for better readability.

B. Preprocessing Text

Since, the text is the most unstructured form of all the available data, various types of noise is present in it and the data is not readily usable without any pre-processing. The entire process of cleaning and standardization of text, making it noise-free and ready for analysis is known as text preprocessing and is a mandatory to get any coherent results.

The main columns we focus our analysis on is the overall column(i.e. rating of the product) and the reviewText column(i.e textual description and product review)

The rating column comes in five categories(range 1-5), essentially 1 to 5 stars, which have been transformed to 3 unique categorical values ranging from 0 to 2, where 0 is for negative reviews, 1 for neutral reviews and 2 for positive review.

The reviewText column storing textual data goes through multiple preprocessing stages before we run it through the model.

1. Removing Punctuation

One important task in text normalization involves removing unnecessary and special characters including punctuation. The main reason for doing so is because although punctuation can provide useful insights into the sentiment of a review, even state of the art language processing models like BERT and GPT3 are unable to pick up on this. The punctuation tends to throw the models off instead of helping them predict the correct class, which is why removing the punctuation is a common practice.

2. Removing stopwords

Stopwords are usually words that end up occurring the most if you aggregated any corpus of text based on singular tokens and checked their frequencies. Articles (a, the ,an), Pronouns (you,them,they,me), Prepositions and so on are stopwords. They have little or no significance. They are usually removed from text during processing so as to retain words having maximum significance and context.

3. Lemmatization and Stemming

The process of lemmatization is reducing the inflectional forms of each word into a common base or root.

Stemming usually refers to a crude process that chops off the ends of words within the hope of achieving this goal correctly most of the time, and sometimes includes the removal of derivational units (the obtained element is understood because the stem).

Alternatively , lemmatization consists in doing things properly with the utilization of a vocabulary and morphological analysis of words, to return the bottom or dictionary sort of a word, which is known as the lemma.

Although Lemmatization seems to be the route to go, the difference that we obtained in the results were insignificant.

4. Tokenization

Tokenization is the process of tokenizing or splitting a string, text into an inventory of tokens. Tokens are subunits of segments of a text document. They can be words, sentences, phrases etc. In our analysis we have used the method of Ngrams for tokenization.

An n-gram may be a contiguous sequence of n items from a given sample of text or speech. The items are often phonemes, syllables, letters, words or base pairs consistent with the appliance .

5. Representing Textual Data in Numerical form

We used two approaches for converting textual data to numerical form so we can feed it to our models. TfidfVectorizer and CountVectorizer both are methods for converting text data into vectors as the models can process only numerical data.

In CountVectorizer (Bag of words) we only count the number of times a word appears in the document which results in biasing in favour of most frequent words. This ends up in ignoring rare words which could have helped us in processing our data more efficiently.

For Term Frequency-Inverse Document Frequency the product of Term frequency and inverse document frequency is used. Term frequency is how frequently a term has appeared in a document. Let's say a term appears f times in a document with d words.

$$\text{Term Frequency} = f/d. \quad (1)$$

IDF is inverse document frequency. If a corpus contains N documents and the term of our interest appears only in D documents then IDF is:

$$\text{IDF} = \log(N/D). \quad (2)$$

TF-IDF is a product of Term Frequency and Inverse Document Frequency. TF-IDF shows the rarity of a word in the corpus. If a word is rare then probably its a signature word for a particular sentiment/information.

VI. EXPLORATORY DATA ANALYSIS

A. Key Relations and Insights from the dataset

1) People are more likely to review a product when they were highly impressed with it.

2) Another observation made is that a customer is more likely to write a review when they find a fault as compared to when they think the product is average.

3) Commonly used words and phrases for good,bad and neutral reviews were analysed to get clarity on what users use to express their sentiment.

B. Visualizations

1) Percentage of reviews in each category

Fig. 2. Comparison of data distribution in various categories of review sentiment.

Referring back to the introduction, the dataset contains more positive reviews as compared to average or bad reviews.

2) Common Phrases used in review according to categories.

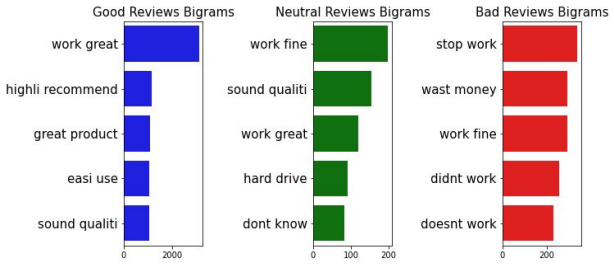
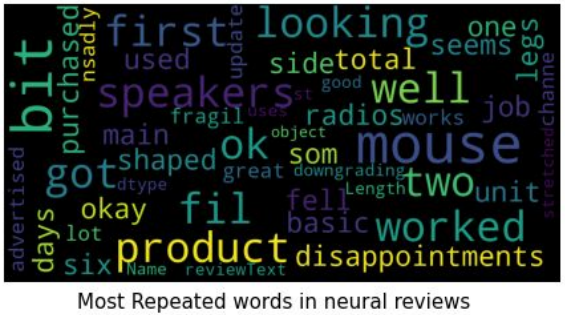


Fig. 3. Different Phrases used in reviews and what sentiment they project.

This depicts the frequency of certain words and phrases occurring in each category of reviews. It is observable that there are product specific phrases like sound quality occuring frequently as the dataset is for electronics, which is dominated by headphones.In negative reviews people tend to point out damaged products, products that fail to work and products that don't match its price value.

3) Word Clouds



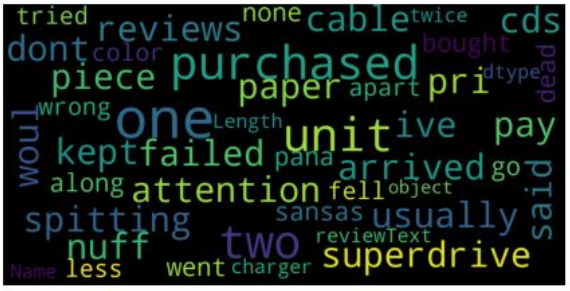


Fig. 4. Comparative study of different word clouds generated.

4) Sentiment Distribution

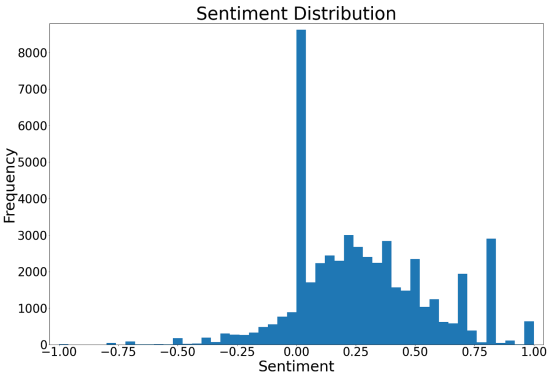


Fig. 5. Variation of word frequency with respect to varying sentiment.

Once again this shows that people tend to leave ratings when they are satisfied with the product.

5) Correlation between review length and sentiment

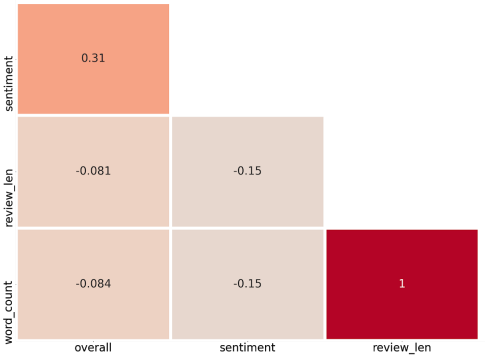


Fig. 6. Correlation heatmap to show varying correlation strength and direction between various data attributes.

Correlation between review length and rating (bad products have longer reviews)

VII. EXPERIMENT RESULTS

In all models implemented below we have implemented selection of k highest scored features of the data to pass through our model.
Further on we have tried to tune parameters of previously executed models and tested it using both CountVectorizer and TF-IDF Tokenizer.

A. NAIVE BAYES [8]

Bayes' Theorem provides a way that we can calculate the probability of a piece of data belonging to a target class using past knowledge. Bayes' Theorem is stated as:

$$P(class|data) = (P(data|class) * P(class)) / P(data) \quad (3)$$

Where P(class|data) is the probability of class given the provided data. Naive Bayes is a classification algorithm for binary (two-class) and multiclass classification problems. Probabilities for each class are simplified i.e they are considered independent of each other.

We worked with three variants of Naive Bayes algorithm - Multinomial Naive Bayes, Complement Naive Bayes and Bernoulli Naive Bayes using (1,1) ngrams and TF-IDF and CountVectorizer.

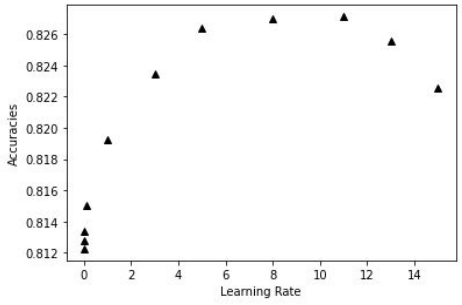


Fig. 7. Graph plotted to best estimate hyperparameters when using CountVectorizer tokenization(Learning rate vs accuracy).

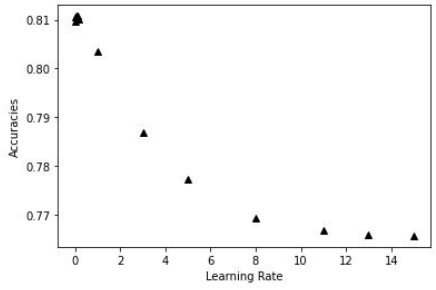


Fig. 8. Graph plotted to best estimate hyperparameters when using TF-IDF tokenization(Learning rate vs accuracy).

TABLE II. NAIVE BAYES ACCURACY RESULTS

MODEL TYPE	**TUNING (k=2890)**	
	***CountVectorizer* Learning Rate :11**	***TF-IDF Learning Rate:0.1***
Multinomial	82.71%	81.07%
Complement	81.15%	78.05%
Bernoulli	75.45%	74.20%

Fig. 9. Comparative study of different variants of Naive Bayes Algorithm

B. RANDOM FOREST CLASSIFIER [9]

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean/average prediction of the individual trees. Most accurate results obtained using the best parameters derived from GridSearch method.

TABLE III. RANDOM FOREST ACCURACY RESULTS

TUNING (k=1200)		
PARAMETERS	***CountVectorizer***	***TF-IDF***
n-estimators : 100 criteria : Gini index No pruning	82.98	83.33
n-estimators : 100 criteria : Entropy No pruning	82.17	N.A.

Fig. 10. Comparative study of different variants of Random Forest Classifier

C. STOCHASTIC GRADIENT DESCENT [10]

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

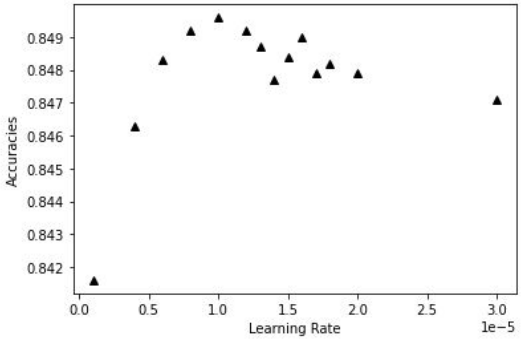


Fig. 11. Graph plotted to best estimate hyperparameters when using Support vector machine model(Learning rate vs accuracy) .

TABLE IV. STOCHASTIC GRADIENT DESCENT ACCURACY RESULTS

LOSS FUNCTION	**TUNING (k value=1800)**		
	PARAMETERS	***CountVectorizer***	***TF-IDF***
Support Vector Machine	Learning Rate : 1e-05 Epochs: 1000	84.14%	84.96%
Logistic Regression	Learning Rate : 1e-05 Epochs: 1000	83.13%	84.74%

Fig. 12. Comparative study of different variants of Random Forest Classifier

D. BERT (Bidirectional Encoder Representations from Transformers) Algorithm

BERT is a model designed by researchers at Google AI Language. It presents state-of-the-art results in a wide variety of NLP tasks. BERT makes use of Transformer which learns contextual relations between words (or sub-words) in a text. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), with the help of Transformers the entire word is read at once.This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word), as opposed to only left context based learning of GPT3. This double sided context is what led us to choose BERT over its competitors as there is no prediction involved, allowing the maximum context to be derived from each item in the dataset

TABLE V. BERT ACCURACY RESULTS

TUNING **Hyperparameters Chosen:** Batch Size : 3 Steps: 0 Learning Rate: 1e-5 Adam Epsilon: 1e-8	
EPOCHS	***ACCURACY***
Epochs:1	88.89%
Epochs:2	89.05%
Epochs:3	89.25%

Fig. 13. Comparative study of different variants of Random Forest Classifier

Due to computational constraints we didn't run further iterations but we believe the accuracy can be increased with more number of epochs.

VIII. RESULTS

After getting the best set of hyperparameters for each of the models we drew a comparison on how they perform with respect to each other to see what model we can expand on in our further studies.

TABLE VI. BEST MODEL VARIANTS AND THEIR ACCURACIES

MODEL	***ACCURACY***
Random Forest Classifier (TF-IDF)	83.33%
Naive Bayes Classifier - Multinomial (Bag Of Words)	82.71%
Stochastic Gradient Descent - Support Vector Machines	84.96%
Stochastic Gradient Descent - Logistic Regression	84.74%
BERT Text Classifier	89.25%

Fig. 14. Comparative study of different ML Models for Text Classification

IX. CONCLUSION

After multiple model iterations and testing, we believe that the BERT classifier does the best job at estimating the sentiment of a review, with an accuracy of almost 90% . Even though the entire testing and analysis was done at a very basic level, we believe that this will be really useful in various fields of product and user relationship analysis. A good application of this would be in recommendation systems, where users can be clustered based on the similar reviews that they give on sites like amazon.

X. REFERENCES

[1] http://jmcauley.ucsd.edu/data/amazon/

[2] Haque, Tanjim & Saber, Nudrat & Shah, Faisal. (2018). *Sentiment analysis on large scale Amazon product reviews.* 10.1109/ICIRD.2018.8376299.

[3] Tim Althoff , Cristian Danescu-Niculescu-Mizil , Dan Jurafsky Stanford University, Max Planck Institute SWS : *How to Ask for a Favor: A Case Study on the Success of Altruistic Requests*

[4] Najma Sultana &Pintu Kumar & Monika Rani Patra & Sourabh Chandra and S.K. Safikul Alam (2019): *Sentimental Analysis of product reviews*

[5] Dublin, Griffith & Joseph, Roshan. (2020). Amazon Reviews Sentiment Analysis: A Reinforcement Learning Approach. 10.13140/RG.2.2.31842.35523.

[6] Dey, Sanjay and Wasif, Sarhan and Tonmoy, Dhiman and Sultana, Subrina and Sarkar, Jayjeet and Dey, Monisha(February 2020) *A Comparative Study of Support Vector Machine and Naive Bayes Classifier for Sentiment Analysis on Amazon Product Reviews*

[7] Najma Sultana &Pintu Kumar & Monika Rani Patra & Sourabh Chandra and S.K. Safikul Alam (2019) : *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*

[8] https://towardsdatascience.com/sentiment-analysis-introduction-to-naive-bayes-algorithm-96831d77ac91

[9] https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[10] https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31