Summarization and Sentiment Analysis of Amazon Reviews

P20, Kaushik Jadhav[1], Shreya Maheshwari[2], Devashish Vachhani[3], Palash Rathod[4]

Department of Computer Science, North Carolina State University, Raleigh, NC 27695

{kajadhav, smahesh4, dvachha, prathod}@ncsu.edu

1. Background

1.1. Introduction and The Problem

There are billions of products on e-commerce sites making it difficult for businesses to classify good and bad products so that they can promote the good products & remove the bad ones to instill customer loyalty and improve customer experience. This problem can be solved using Sentiment Analysis. Sentiment Analysis is an application which comes under the domain of Natural Language Processing (NLP), wherein, human sentiment is detected and analyzed based on speech or text. There are a plethora of ways for companies to get data for analyzing the customer sentiment. Apart from direct customer interactions, customers also rate and review products. Machine learning models can be trained with such data and a general trend of positive, neutral or negative sentiment can be analyzed for the same. This positive, negative or neutral sentiment will help e-commerce vendors determine how customers are feeling about a particular product and likewise promote or demote it from their website.

1.2. Literature Survey

It has been proven in previous studies that applying proper tokenization and pre-proecessing to reviews and feeding them to neural networks can predict sentiment in reviews [1]. There are also older studies that talk about using LSTM for review analysis [2]. However, those are without an embedding layer, unlike the one we built in this project. In [3], the authors have articulated how to perform Hyperparameter tuning and Cross Validation for Amazon reviews models. Also, recent studies show how we can employ previously unused supervised models like Logistic Regression for review Sentiment Analysis [4]. In this project, we aim to perform sentiment analysis using 4 Supervised Machine Learning models, namely, Multinomial Naive Bayes, K-Nearest Neighbour (KNN) Clustering, Logistic Regression and Decision Trees. Further, we explore Sentiment Analysis using Deep Learning with LSTM model and compare it's performance with the aforementioned machine learning models.

2. Proposed Method

2.1. Intuition

The state of the art method used by ecommerce companies for analysing customer reviews only involves plotting graphs and pie charts of star ratings. Although such plots are a good way to visualize star ratings, they do not derive any insights from the customer comments and the emotions of the customer who wrote the review. Of course, there are

a few other similar studies where people have used machine learning models like Gaussian Naive Bayes, KNN, Logistic Regression, LSTM and SVM for sentiment analysis of customer reviews [1,2,4]. But none of the studies so far have ever evaluated Decision Tree & Multinomial Naive Bayes models. Also, the LSTM models used in earlier studies either do not have an Embedding layer, or have one with fewer words. In our project, the Embedding layer takes 20000 words at a time from the corpus at each iteration. So in this project, as we are training previously researched models (KNN, Logistic Regression) along with our newly researched models (Decision Tree, Multinomial Naive Bayes, Enhanced LSTM), we should get better insights compared to the state of the art method.

2.2 Description of Algorithms

The original dataset is raw and unclean. So, we first apply data pre-processing, transformation and modeling and then train models on it. Further we also perform Hyperparameter Tuning and Cross Validation. These would be discussed in more detail in Section 3. The semantics of the algorithms used are listed below.

Multinomial Naive Bayes

Naive Bayes deals with probability as the "likelihood" that data belongs to a specific class. The basic idealogy behind a Naive Bayes model is the assumption that all the features are conditionally independent of one another given some class. A major advantage of a Naive Bayes classifier is that its computation time is extremely fast however the Naive Bayes assumption of independence is rarely true.

Decision Tree

As the name suggests, a decision tree classifier uses a tree-like structure to model decisions. Each internal node denotes an attribute, each branch represents an outcome, and each leaf node holds a class label. The construction of a decision tree classifier does not require any domain-specific knowledge and therefore is appropriate for exploratory discovery. Generally, a decision tree classifier gives good accuracy.

Logistic Regression

Logistic Regression is a popular binary classification algorithm. The term 'logistic' is derived from the function which is used at its crux known as the logistic or the sigmoid function. This function maps any real valued number to a value between 0 and 1, excluding the limits. Its works as a classifier when a decision threshold is introduced.

K-Nearest Neighbour

The K-Nearest Neighbour (KNN) algorithm is a supervised machine learning algorithm used for estimating the likelihood that the point will become a member of one group or another depending on the labels of the data points near it. It is a lazy learner as the model is not created beforehand, but only constructed when the test data is provided. For each point in the testing data, we find the distance of all points in the training data. Then these are sorted and the top k distances are taken. The label for the test point is based on the majority of classes presented in the selected points.

Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) is a Deep Learning Neural Network that follows a layered approach. The first layer in our neural network is an Embedding layer that which

takes 20000 rows as input and produces embeddings of shape 128. This 128 shape output is given to an LSTM layer with corresponding number of neurons. Finally, we have our final dense layer with Softmax activation function and the loss function used is loss function is binary_crossentropy. The final output layer has 2 neurons as we have two output classes: 1 for positive sentiment, 0 for negative sentiment. LSTMs are highly efficient for learning patterns in unstructured data like the customer reviews in our case.

3. Plan & Experiment

3.1. Dataset

The dataset used is Amazon Kindle reviews from Kaggle. Before cleaning, there are 982618 reviews present in the dataset given from from May 1996 - July 2014. The columns present are:

- asin: ID of the product
- helpful: helpfulness rating of the review
- overall: rating of the product
- reviewText: text of the review (heading)
- reviewTime: time of the review (raw)
- reviewerID: ID of the reviewer
- reviewerName: name of the reviewer
- summary: summary of the review
- unixReviewTime: unix timestamp

3.2. Hypothesis

Our hypothesis is when Deep Learning models like LSTM are enhanced by an embedding layer, do they outperform Machine Learning models? Another hypothesis is whether the performance of new models which were never used for review sentiment analysis like Decision Tree and Multinomial Naive Bayes were better than models used in previous studies? We would also like to test how much impact hyperparameter tuning and cross validation have on the previously obtained raw accuracies?

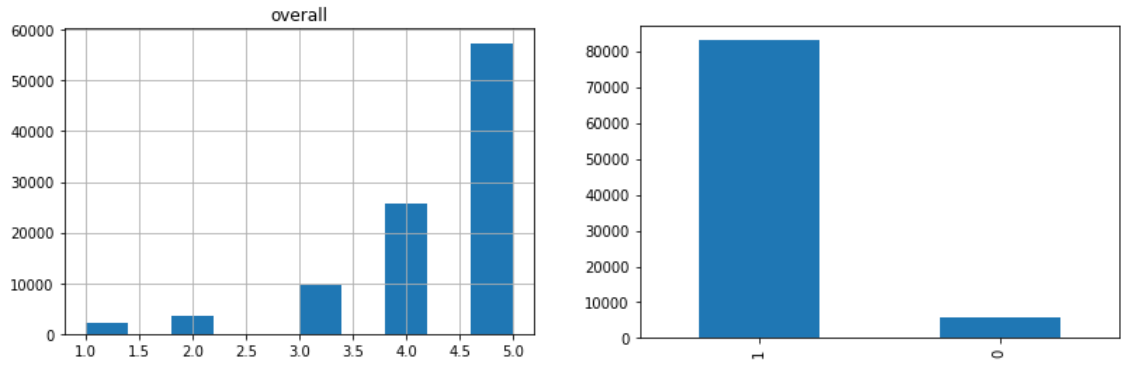
3.3. Experimental Design

3.3.1. Preprocessing

Due to limitation in computational power, only a subset of the dataset is consdiered - 10% of the rows. The following steps are taken to obtain clean data on which analysis can be performed:

- Handling missing values: In some of the rows, reviewText and summary have missing values. Since these are few in number, compared to the size of the dataset, they are dropped.
- Drop redundant columns: Since reviewerID is present for every row, we do not require reviewerName.
- Stemming: It is a normalization technique where list of tokenized words are converted into shortened root words to remove redundancy. This is done by making use of PorterStemmer from the nltk library.
- Lemmatization: It gives the form of the related word in dictionary order. This is done by making use of WordNetLemmatizer.

3.3.2. Data preparation and transformation



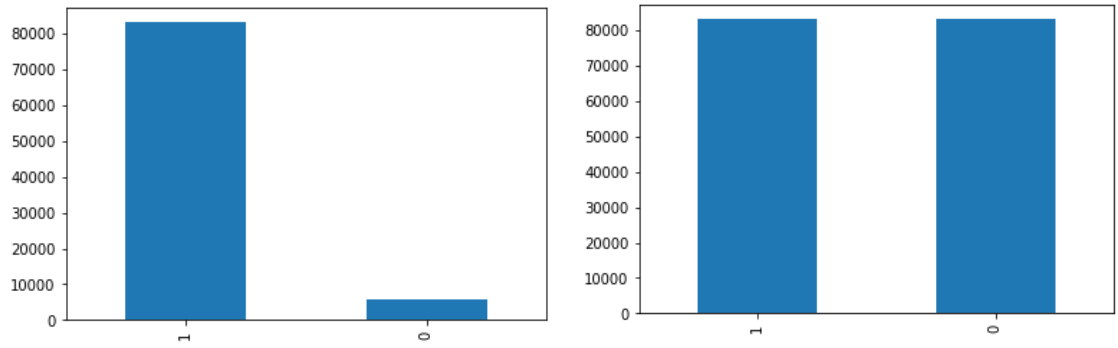
From the graph, we can observe that there are 5 unique ratings in the dataset. In order to label each score with a sentiment, we decide to split the range of ratings equally.

- All reviews with rating 1 and 2 - negative, label: 0
- All reviews with rating 3 - neutral, these reviews are removed
- All items with rating 4 and 5 - positive, label: 1

The data present under the reviews column is in the form of strings. Machine learning models take numbers as input; therefore, these strings will need to be converted to vectors. This is done by utilizing CountVectorizer.

3.3.3. Data sampling

The distribution of reviews with the two labels is shown in the figure on the left. From the graph, it can be seen that the data is extremely unbalanced. In order to ensure equally balanced data, RandomOverSampler is used. From the figure on the right, it can be seen that the distribution of reviews with the two labels is comparable now.



3.3.4. Exploratory Data Anlaysis

A word cloud is constructed to graphically depict which words have higher frequency in the dataset. The word clouds below show the words occurring with high frequency in reviews with positive sentiment and negative sentiment respectively.

3.3.5. Classifier Experimentation

After preparing the data, 90% is used for training and 10% is used for testing. Through the experiment, our aim is to use the training data to train different Machine Learning models and find the accuracy on the test set. In order to ensure that the model is generalized and works well for new and unseen data, 5-fold cross validation is performed. In each split, 4 folds of the data are used for training and the remaining 1 fold is used for testing. However, k-fold cross validation selects data by random sampling. So, there might be a case where all points in the testing set of a fold belong to one class. In order to avoid this, stratified k-fold cross validation is used, because it selects data by stratified sampling, which ensures that data of both classes is uniformly distributed in each set. Through hyperparameter optimization, we find a set of optimal hyperparameter values for a learning algorithm while applying the algorithm to any dataset. Thus, for further improvement in accuracy, hyperparameter optimization will be performed on all the models. Using the optimal parameters, accuracy will be calculated again.

4. Results

4.1. Metrics

For evaluating different classifiers, we have used the following metrics:

- Accuracy
- Precision
- Recall
- F-1 Score

Model	Hyperparameters	Accuracy	Precision		Recall		F-1 score	
			0	1	0	1	0	1
Multinomial Naive Bayes	alpha: 1.0	0.94	0.92	0.96	0.96	0.92	0.94	0.94
Decision Tree	criterion: gini, min_samples_split: 2	0.96	0.93	1	1	0.93	0.97	0.96
Logistic Regression	penalty: l2, C: 1.0, solver: liblinear	0.97	0.96	0.99	0.99	0.96	0.98	0.97
K-Nearest Neighbors	neighbors: 1	0.98	0.95	1	1	0.95	0.98	0.98
Enhanced LSTM	loss: binary_crossentropy, optimizer: adam	0.97	0.96	0.99	0.99	0.96	0.98	0.97

Tabela 1. Metrics

4.2. Plots

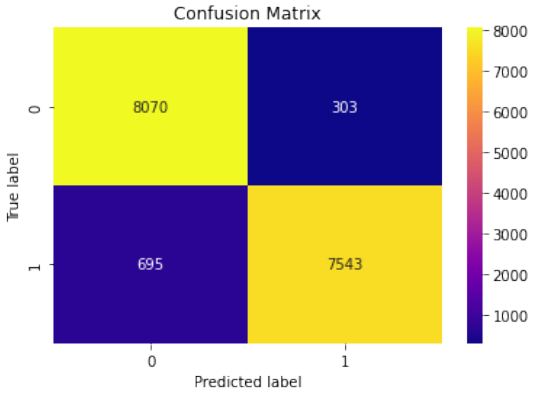


Figura 1. Naive Bayes

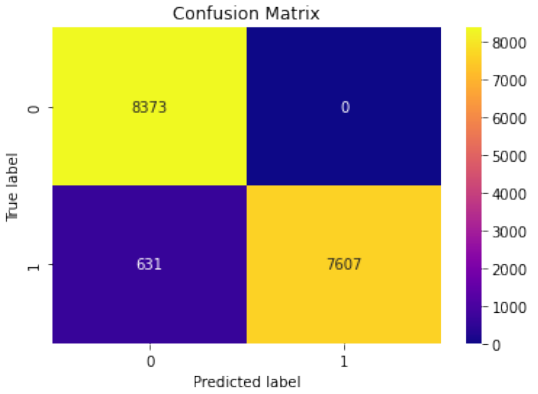


Figura 2. Decision Tree

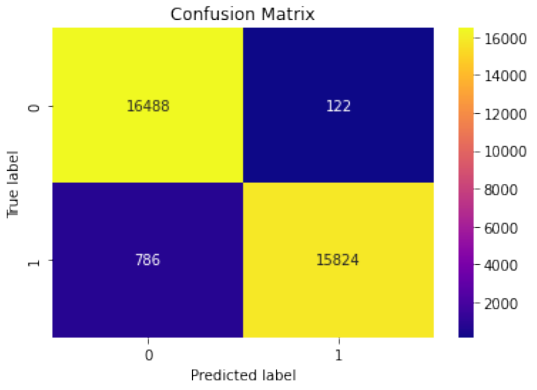


Figura 3. Logistic Regression

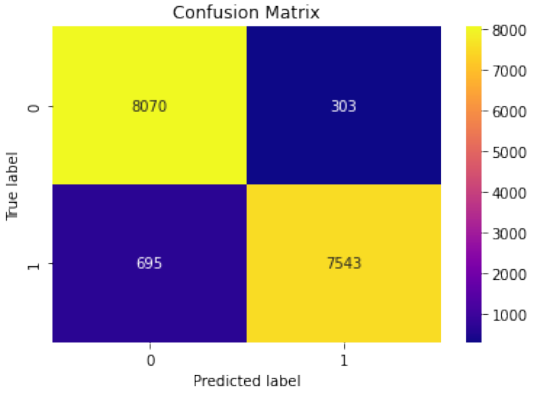


Figura 4. K-Nearest Neighbours

4.3. Hyperparameter tuning

Model	Hyperparameters	Accuracy	Precision		Recall		F-1 score	
			0	1	0	1	0	1
Multinomial Naive Bayes	alpha: 0.0001	0.95	0.94	0.97	0.97	0.94	0.95	0.95
Decision Tree	criterion: entropy, min_samples_split: 2	0.97	0.94	1	1	0.94	0.97	0.97
Logistic Regression	penalty: l2, C: 1.0, solver: liblinear	0.97	0.96	0.99	0.99	0.96	0.98	0.97
K-Nearest Neighbors	neighbors: 1	0.98	0.96	1	1	0.96	0.98	0.98
Enhanced LSTM	loss: binary_crossentropy, optimizer: adam	0.98	0.96	0.99	0.99	0.95	0.98	0.97

Tabela 2. Hyperparameter tuning

4.3.1. ROC Curve

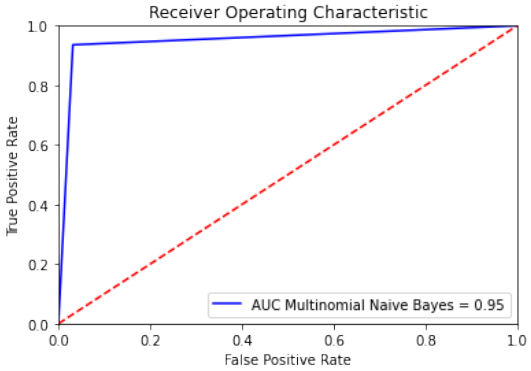


Figura 5. Naive Bayes

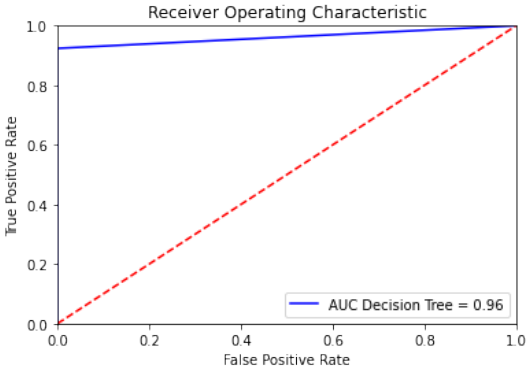


Figura 6. Decision Tree

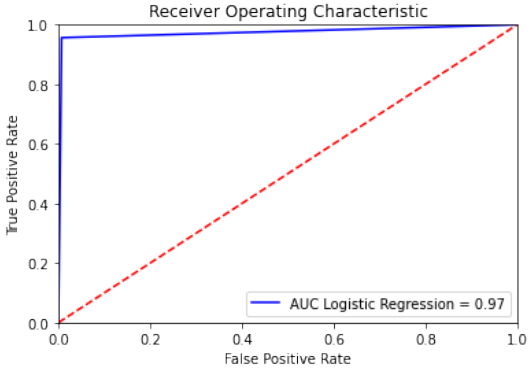


Figura 7. Logistic Regression

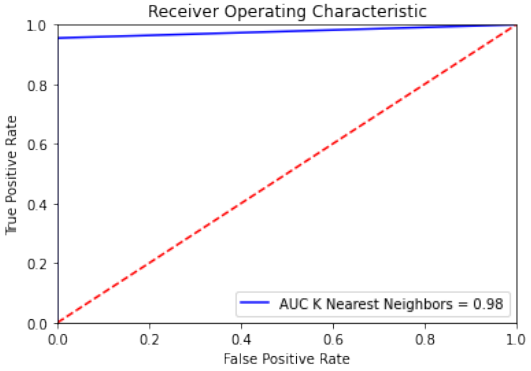


Figura 8. K-Nearest Neighbours

4.4. Stratified K-fold cross validation

Model	Accuracy	Precision		Recall		F-1 score	
		0	1	0	1	0	1
Multinomial Naive Bayes	0.95	0.93	0.97	0.97	0.93	0.95	0.95
Decision Tree	0.97	0.94	1	1	0.94	0.97	0.97
Logistic Regression	0.97	0.95	0.99	0.99	0.95	0.97	0.97
K-Nearest Neighbors	0.97	0.95	1	1	0.95	0.98	0.97
Enhanced LSTM	0.98	0.96	0.99	0.99	0.95	0.97	0.97

Tabela 3. Stratified K-fold cross validation

4.5. Critical Evaluation

To begin with, the testing accuracy is a good measure of performanace since we balanced the data in the pre-processing step and the cost of false-positive and false-negative is the same. As expected, the baseline models Logistic Regression and K-Nearest Neighbour

performed exceptionally well on the dataset and provided a testing accuracy of 97% and 98% respectively.

Coming to the first question of the hypotheses, we observe that the Long Short Term Memory (LSTM) model outperforms the baseline models as well as the novel machine learning models with a testing accuracy of 98.1%. This can be explained by the fact that deep learning models make an attempt to understand the intricacies of the reviews (which is not done by the machine learning models) and thus map the sentiment score better.

For the second and third question of the hypotheses, both novel machine learning models Multinomial Naive Bayes and Decision Tree also performed comparably to our baseline models with a testing accuracy of 94% and 96%. With hyperparameter tuning using **GridSearchCV**, we were able to further improve this to 95% and 97%.

5. Conclusion

This study was successful in forming a comparative study of machine and deep learning models for sentiment analysis. As per the results, LSTM performed better than the machine learning algorithms. However, it will not be right to generalize it for all tasks. There are many factors that have to be considered further. Firstly, this study was based on a subset of a dataset. With more computational resources, we can train these algorithms for better performance. Also, the scope of this study can be widened by incorporating different algorithms to get a better view. There are also many pre-trained models such as Bert which can be fine tuned as per our requirements and used accordingly.

Another observation made during this study was how the hyperparameters should be chosen. For certain algorithms, not all hyperparameters are compatible with one another. Thus while optimizing them, we need to carefully choose a subset of hyperparameters that are compatible. In order to choose the subset of hyperparameters, we need to assess the problem statement and accordingly choose them.

A. References

[1] Huiliang Zhao, Zhenghong Liu, Xuemei Yao, A machine learning sentiment analysis of product reviews, Information Processing & Management, Volume 58, Issue 5, 2021

[2] Onan, Aytuğ, Sentiment analysis on product reviews based on word embeddings and neural networks, Concurrency and Computation: Practice and Experience, 2021

[3] Wassan, Sobia & Chen, Xi & Shen, Tian & Waqar, Muhammad, Zaman, Noor. (2021). Amazon Product Sentiment Analysis using Machine Learning Techniques, 695-703

[4]Wankhade, Mayur & Chandra, A & Rao, Sekhara & Dara, Suresh & Kaushik, Baij. (2017). A Sentiment Analysis of Food Review using Logistic Regression. 2456-3307.

[5]Murthy, Dr & Allu, Shanmukha & Andhavarapu, Bhargavi & Bagadi, Mounika. (2020). Text based Sentiment Analysis using LSTM.

B. Project Link

https://github.ncsu.edu/kajadhav/engr-ALDA-Fall2022-P20