University of Maryland – Baltimore County

Master's of Professional Studies: Data Science

Project

Amazon Product Reviews: Sentiment Analysis and Star Rating Prediction

Abhinay Nagulapalli, Devi Prasanna Sai Suresh Takasi, Gauree Pramod Khapekar, Neha Dave, Sai Kumar Reddy Gheereddy

DATA 602 Intro to Data Analysis and Machine Learning

Prof. Tony Diana

Fall 2020

Abstract: In the present era, customer opinion is vital for a consumer brand to succeed. This data can be considered as gold mines as they are invaluable and contain lots of underlying information about customer sentiments. People tend to purchase online based on the reviews and ratings of the product making the reviews as important as personal recommendations. In the project, these reviews are pre-processed, analyzed using Natural Language Processing or text analytics techniques and determine how these textual reviews have justified the star ratings. The features that are derived in the analysis from textual reviews are used to classify its corresponding star ratings. The problem is transformed into a multi-class classification task to classify reviews on the basis of its star rating from one of the five classes. The performances of various classifiers used are evaluated and compared. The aim of this paper is to provide the insights of the project that has been accomplished in order to analyze the sentiments of the customer reviews and the ratings of the products for the eCommerce site of Amazon.

Introduction: With the increasing pace of technology and in the digital era almost everything can be purchased online. Public opinion about buying the products online is the primary way to develop the trust for buying those products as the purchasing decision of new potential customers in absence of true feel of the product depends on the reviews of existing customers. It is crucial to analyze people's sentiments. This analysis particularly is helpful when it comes to negative reviews as it is helpful to discover the shortcomings. Ratings are so much built into our psyches that the trust is not an obstacle with them anymore. We believe in them subconsciously, we take trade decisions based on them, and we genuinely invest in them with our personal choices (Mukherjee, 2020). However, the overall star ratings of the product reviews may not often capture the accurate polarity of the sentiments. Also,

customers assign these ratings according to their reviews. Thus reviews and ratings should go hand in hand. But this leads us to an important research question: Do reviews and ratings affect each other or play independent roles? Is it true that higher star rating means positive sentiment and lower star rating means negative sentiments? This project's one of the aims is to understand if we can find out star rating if we have review text available. The second aspect is about understanding the sentiment of review text which is helpful to the companies for product and service improvements, understanding customer needs and making strategies for business development. The rest of the paper is organised into sections. The next section discusses the entire approach of the study. This is followed by the experimental results, conclusion and future scope.

Approach: This section is dedicated to the flow of the project. All the steps taken to build the classification model and understanding the concepts are explained in detail.

Dataset collection and its features: Amazon Customer Reviews is one of Amazon's iconic products making it a rich source of information for academic researchers in the fields of Natural Language Processing, Information Retrieval, and Machine Learning, amongst others (amazon). The chosen dataset contains product reviews of Mobile electronics purchased from Amazon.com. It includes 104852 rows and 15 features as explained below. The data obtained as the gzip file is extracted and explored. Each row corresponds to a customer review, and includes the feature variables:

marketplace	Marketplace of the product
customer_id	ID of the reviewer
review_id	ID of the review
product_id	ID of the product
product_parent	ID of the product parent
product_title	Product title
product_category	Category of product

star_rating	Rating of the product
helpful_votes	Helpful votes of the review
total_votes	Total votes of the review
vine	Indicator for vine review
verified_purchase	Indicates if the purchase is verified or not
review_headline	Customer review title
review_body	Customer review body
review_date	customer review date

Data Exploration and Cleaning: Exploring the data set and developing a deep understanding of the data is one of the most important tasks we have done as part of the project. We used Matplotlib, Seaborn, and Pandas to perform data exploration. As part of the Data Cleaning in our case, we dropped the rows where some information was missing especially related to review text, and star rating as those are important features as part of this project is concerned. This led to the dropping of 8 rows which constituted a very small fraction of the whole dataset. On the other hand, we also derived the day, month and year from the review_date column to be used in further analysis.

Data Visualization: Data visualization is essential for understanding the dataset by identifying patterns and trends that might go undetected in text-based data (Umer, Ashraf, Mehmood, Ullah, & Choi, 2020). We plotted the distribution of all the numerical features to understand the data. The star rating distribution in Figure 1 shows that there are many more 5-star ratings than other ratings. Hence, it is important that we take this into account when creating our training and testing sets or we may end up with too many 5-star reviews in one set.

Figure 1: Distribution of Star Rating

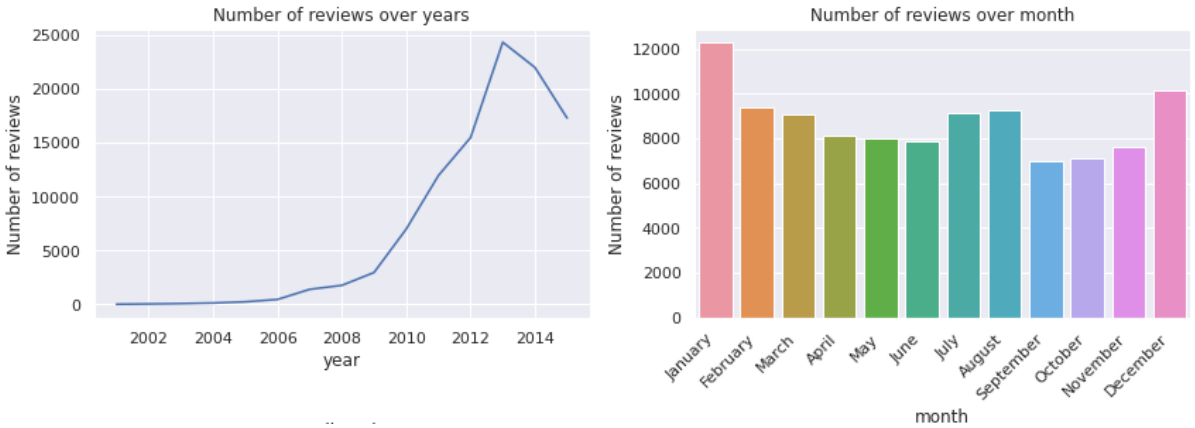


Figure 2: Trend over the years and months

Rating trend over the years in Figure 2 shows that there is an increasing pattern in the number of ratings given by the users to products on Amazon which indicates that a greater number of users started using the Amazon e-commerce site for online shopping and a greater number of users started giving feedback on the products purchased from 2006 to 2014. There is a significant increase in the number of ratings given by users between 2012 and 2013. We also noticed a peak in 2013 and a major event that supports these findings. Amazon began to

offer Sunday delivery options for purchases which surely resulted in an increasing number of members, ratings & reviews.

Review length distribution shows that the 2-star, 3-star and 4-star reviews have a comparatively greater length. This might be because people tend to give very long reviews when they are explaining details about the product. For example, if someone doesn't like a particular feature of some product they explain it in detail.

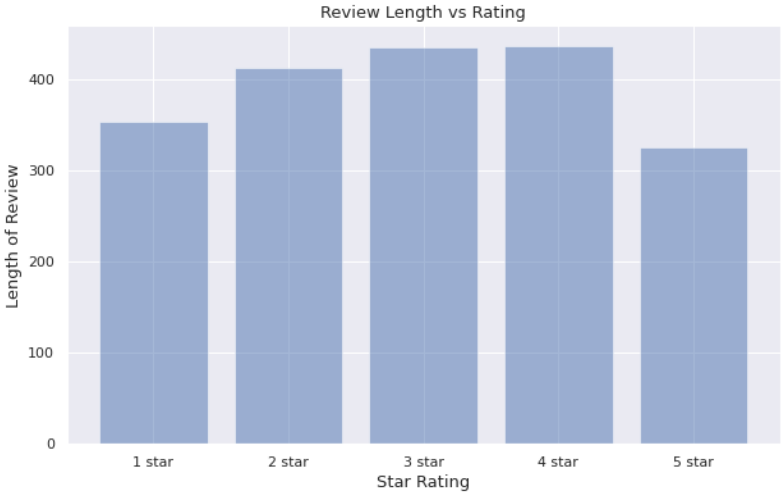


Figure 3: Review Length vs Rating

Review Polarity & Subjectivity: Polarity in sentiment analysis refers to identifying sentiment orientation in written or spoken language. The TextBlob package for python is a convenient way to do a lot of Natural Language Processing (NLP) tasks (TextBlob Sentiment: Calculating Polarity and Subjectivity 2015). Polarity lies in the range of [-1,1] where 1 means positive statement, -1 means a negative statement and 0 can be determined as a neutral statement. The polarity has been calculated for each review of the dataset. It is also calculated against each user rating in the range of 1 and 5 for the respective product. From Figure 4, we can see that neutral statements (0) are more in the dataset than the most positive statements (1). Comparatively, there are very few negative statements (Falling below 0) in the dataset. From the Figure 5 box plot, the weight of the positive ratings i.e., above 4 or 5 is mostly distributed

over positive polarity scores. Whereas the neutral rating 3 weight is at the neutral polarity and most of the ratings 1 and 2 have negative statements or reviews which look more obvious.

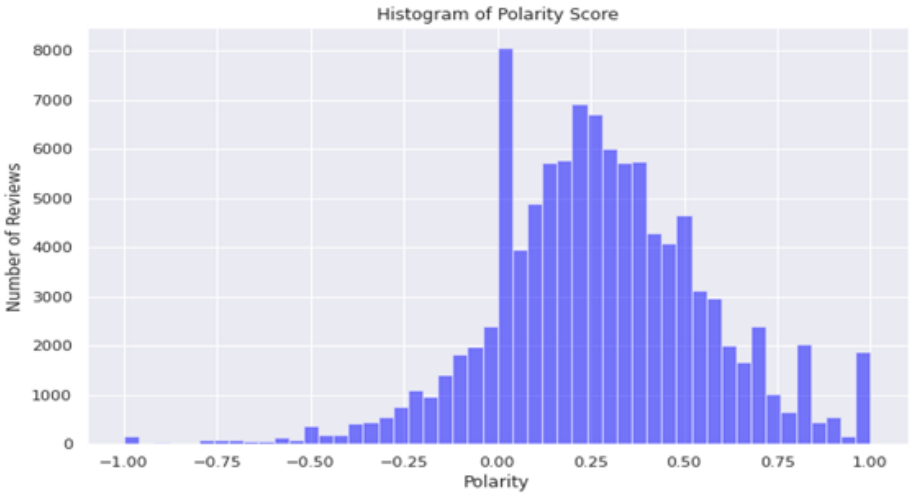


Figure 4: Distribution of polarity scores for all the reviews

Figure 5: Star ratings vs Polarity scores

Subjective sentences express some personal feelings, views, or beliefs. Subjective sentences generally refer to personal opinion, emotion or judgment whereas objective refers to factual information. Subjectivity lies in the range of [0,1]. When it is close to 0, it is more about facts. When subjectivity increases, it comes close to being an opinion. In the data set, the

distribution of subjectivity scores for the reviews are similar to a normal distribution (Figure 6).

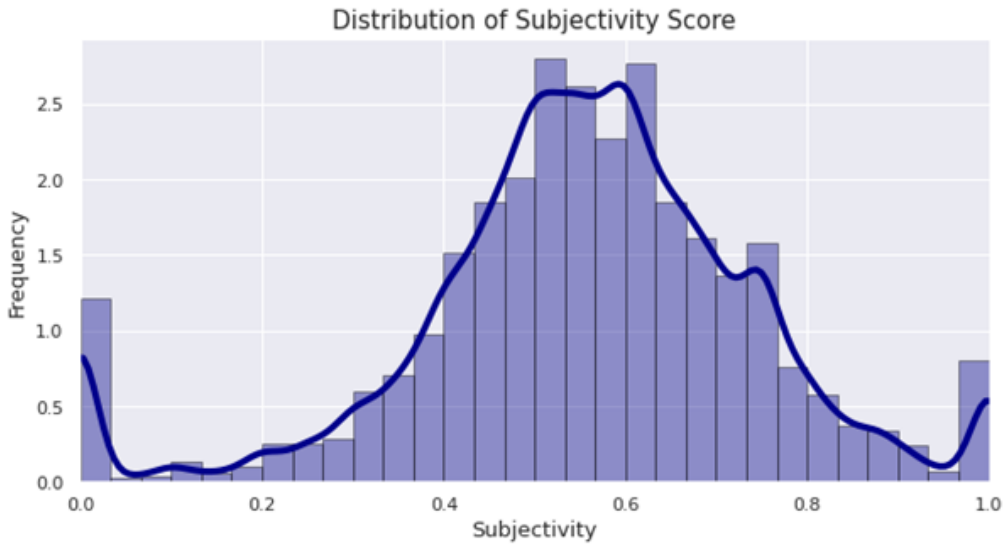


Figure 6: Distribution of Subjectivity score

Emotion Detection: Emotion Detection from text is useful in many ways. It helps to understand the customers, analyze feedback and reviews. Text2Emotion is the python package developed with the clear intention to find the appropriate emotions embedded in the text data. The get_emotion() method of the text2emotion package, processes any textual data, recognizes the emotion embedded in it and provides the output in the form of a dictionary with 5 basic emotion categories such as Happy, Angry, Sad, Surprise, and Fear (Band, 2020). The get_emotion method has been applied to the textual customer reviews of the dataset to get the corresponding dictionary of emotions for each review. From Figure 7, we can see that ratings in the range of 1 and 3 have more surprise and sad emotions in their reviews and ratings of 4 and 5 have more happy emotions in the review. Surprisingly, there is more fear emotion in all the users when writing the review for the product. The emotion tone has been determined based on the get_emotion() method, returned dictionary of emotions.

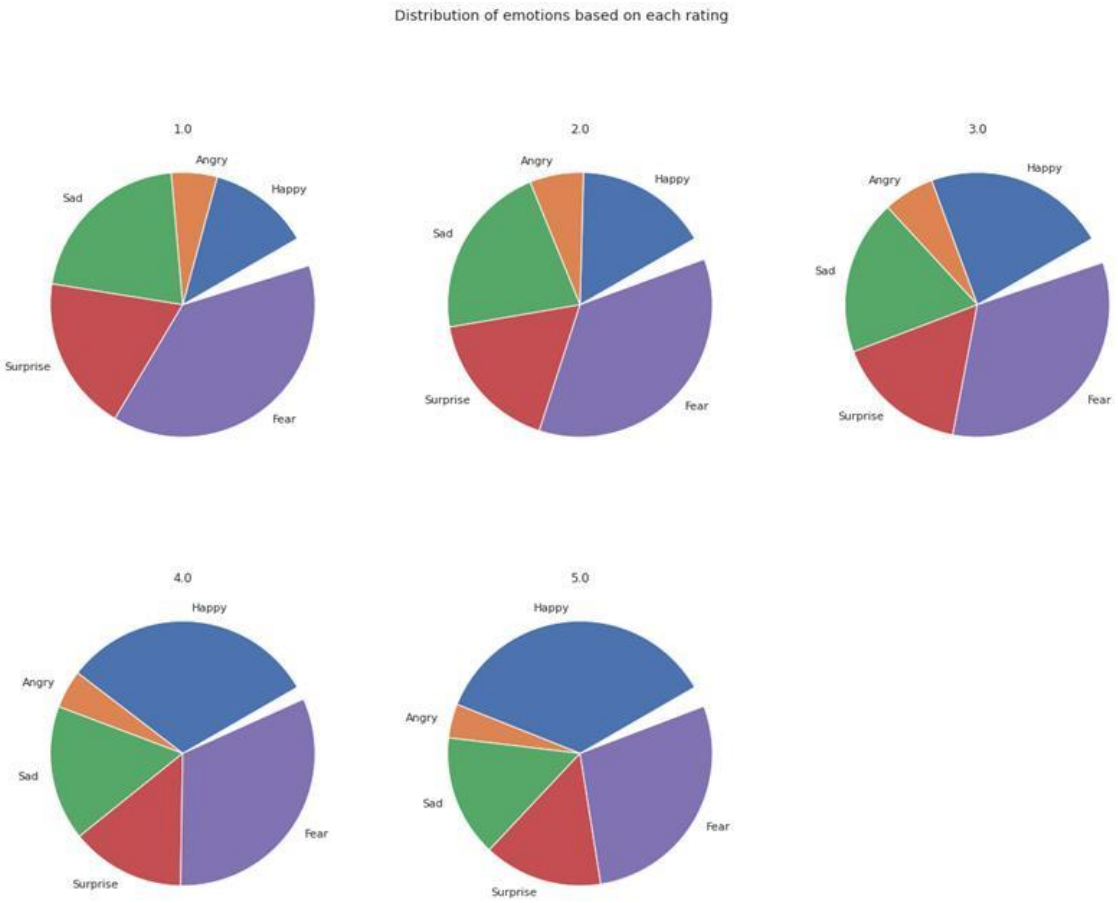


Figure 7: Pie charts showing emotions for each star rating

Sentiment Analysis using VADER: Sentiment Analysis is the automated process of understanding the sentiment or opinion of a given text. VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that uses a combination of a sentiment lexicon as a list of lexical features (Rai, 2019). VADER not only tells about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is. We generated the overall positive/neutral/negative sentiment of the review. The sentiment of the review should have a high correlation with the star rating. The intuition to generate this feature is fairly obvious. It is logical for us to assume that a 1-star review will contain more negative words and hence have a more overall negative sentiment than a 5-star review. We also calculated the sentiment of each review along with the overall sentiment of

the dataset. We observed a very high Neutral sentiment in that dataset. This can be explained by people subtly expressing their positive/negative opinions which could not be captured by the NLTK Sentiment Analyzer due to the Random Ordering of Words.

Text Pre-processing: Since large datasets require longer training times, and because stop words reduce the prediction accuracy, text pre-processing is therefore required to overcome this limitation (Umer, Ashraf, Mehmood, Ullah, & Choi, 2020). This text pre-processing includes various tasks like removing HTML tags, removing URLS, removing accented characters and removing extra characters. This also includes removing punctuations, special characters as they do not contribute to sentiment which leaves only alphabetic and numeric characters. Further converting to lowercase so the same words are recognised as the same. Apostrophes connecting words are used everywhere, especially in public reviews. To maintain uniform structure, it is recommended they should be converted into standard lexicons (Jones, 2018). Lemmatization is the process that finds the base or dictionary form of the word known as the lemma. This is done through the use of vocabulary (dictionary importance of words) and morphological analysis (word structure and grammar relations) (Jones, 2018). Involving all the above steps we cleaned the textual reviews. For this, we have used spaCy which is an open-source software library for advanced Natural Language Processing, written in the programming languages Python and Cython. Research has shown that text pre-processing plays a significant role in improving the prediction accuracy. These pre-processing steps help convert noise from high dimensional features to the low dimensional space to obtain as much accurate information as possible from the text (Umer, Ashraf, Mehmood, Ullah, & Choi, 2020). In order to process the text pre-processing faster, we used Swifter that is available on pip from the swifter package. It makes it easy to apply any function to your pandas series or dataframe in the fastest available manner (Carpenter, 2020).

Topic Modelling using LDA: Online reviews are quite often overwhelming in terms of numbers and information, an intelligent system, capable of finding key insights (topics) from these reviews, will be of great help for both the consumers and the sellers(Mining Online Reviews using Topic Modeling with NLP 2019). Firstly, it enables consumers to quickly extract the key topics covered by the reviews without having to go through all of them. Secondly, it helps the sellers/retailers get consumer feedback in the form of topics (extracted from the consumer reviews). To solve this task, we used the concept of Topic Modeling (LDA) on Amazon mobile electronics review data. Topic Modeling is a process to automatically identify topics present in a text object and to derive hidden patterns exhibited by a text corpus (Joshi, 2020). In our case, instead of text documents, we have thousands of online product reviews. Since topic modeling is an unsupervised learning algorithm, it doesn't require any training. It learns on every iteration it goes through, so it is quick and easy way to start analyzing the data. But a major concern with using topic modelling is, it doesn't guarantee accurate results. Our aim here is to extract a certain number of groups of important words from the reviews. These groups of words are basically the topics which would help in ascertaining what the consumers are talking about in the reviews (Joshi, 2020). This helped us find some hidden topics over all the reviews. Our results included topic (topic 1) which had terms like 'charge', 'charger', 'battery' indicating that the topic is very much related to phone charging. Similarly, another topic (topic 8) is about the overall value of the product as it has terms like 'excellent', 'great', and 'recommend'. We also used the pyLDAvis library to visualize our topics in a 2-dimensional space. This visualization is interactive in nature and displays topics along with the most relevant words.

Named Entity Recognition: Named-entity recognition (NER) (also known as entity identification, entity chunking and entity extraction) is a subtask of information extraction

that seeks to locate and classify named entity mentions in unstructured text into predefined categories such as the person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc (Menzli, 2020). We performed entity recognition on most helpful reviews and also on products with the highest number of reviews. Additionally, we used Displacy to find the entities in the text. NER helped us to get a glance, understand the subject or theme of the reviews and quickly group based on their relevancy or similarity.

Techniques and Algorithms: Training a supervised machine learning algorithm requires textual documents to be represented in vectorial form. For this purpose, textual data must be converted into numbers without losing information (Umer, Ashraf, Mehmood, Ullah, & Choi, 2020). We are going to use term frequency–inverse document frequency (TF-IDF) as our feature extraction strategy. This involves counting the words in a document to produce a matrix displaying the total number of occurrences of each word in the document. TF-IDF takes into account document length and term frequency when building features. We don't want long reviews to seem more relevant just because they are long, we also don't want to overvalue words that appear many times but do not provide much information.

TF/IDF = Term Frequency × Inverse Document Frequency

TF-IDF uses a bag of words with weights to represent a document. However, this will miss some relationships between words. For example "not bad" and "very bad" both contain the word "bad" but one has the inverse meaning. We can help fix this problem using the ngram_range option in sklearns implementation of TD-IDF. ngram_range allows us to set the number of words that can be considered together.

Logistic Regression: Logistic regression is a classification algorithm that takes a vector of features and transforms it into a probability by feeding it into a sigmoid function. Given a set of training points logistic regression will try to find a vector of weights that when multiplied to the set of features and fed into the sigmoid function we get an accurate prediction. Even though logistic regression can only handle binary classification problems, sklearns implementation gives us the option to use a One-vs-Rest technique. This makes logistic regression a linear model capable of multiclass classification.

MultinomialNB: MultinomialNB is a naive Bayes classifier. Naive Bayes algorithms are used extensively in text classification problems. They are highly scalable, in fact, MultinomialNB is O(n) (n=training samples size) in training time. Naive Bayes algorithms are based on Bayes rule and training MultinomialNB is fast.

LinearSVC: The LinearSVC method applies a linear kernel function to perform classification and it performs well with a large number of samples (3.2.4.1.5. sklearn.linear_model.LogisticRegressionCV). LinearSVC is also one of the algorithms which perform quite well on a range of NLP based text classification tasks. Linear SVC provides a decision_function method. The *decision_function* predicts the confidence scores for the samples. The confidence score for a sample is the signed distance of that sample to the hyperplane (Saxena, 2018).

Random Forest: A forest is composed of trees. It is said that the more trees it has, the more robust a forest is. Random forests create decision trees on randomly selected data samples, get a prediction from each tree, and select the best solution by means of voting. Random forest is considered as a highly accurate and robust method because of the number of decision trees participating in the process. It does not suffer from the overfitting problem.

The main reason is that it takes the average of all the predictions, which cancels out the biases. However, the random forest is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming (Raj, 2020).

SGD classifier: SGDClassifier is a very efficient algorithm that uses stochastic gradient descent to build a model. SGDClassifier isn't the model itself, it's a method of building a model. We can specify which model we want to build using the loss parameter. SGDClassifier tries to find the minima or maxima of the loss function by iterating over random samples from the training data. This is faster than training over all the training data. This makes SGDClassifier scalable, and a perfect candidate algorithm for our problem.

Implementation: The experiment is conducted on 104847 customer reviews from the dataset. The reviews are analyzed and pre-processed using our proposed framework. 80% of the reviews are used for training and rest 20% are used for testing the models. Multi-class classifier models used include Multinomial Naïve Bayes Classifier, Logistic Regression Classifier, Random Forest Classifier, SGD (Stochastic Gradient Descent) and Linear SVC (Support Vector Classifier). These models are trained and utilized to classify reviews to one of the 5 classes using TF-IDF vector features created from review text.

We also performed parameter tuning using sklearn's GridSearchCV algorithm. GridSearchCV does an exhaustive search over specified parameter values.

MultinomialNB: MultinomialNB has an alpha hyperparameter that we set using GridSearchCV. The alpha parameter determines what value we give when we see a feature that we haven't encountered in the testing data.

```
parameters = { 'alpha': [0.1, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2] }
nb_clf = GridSearchCV(MultinomialNB(), parameters)
nb_clf.fit(X_train, y_train)
```

SGDClassifier: SGDClassifier has many hyperparameters to play with; we tuned parameters such as alpha, penalty, and loss. The alpha parameter is a constant that multiplies the regularization term, the penalty parameter is the regularization term, and loss is the loss function (the model to train).

```
parameters = [{ 'loss': ['hinge', 'log', 'perceptron'],
                'alpha': 10.0**-np.arange(1,7),
                'penalty': ['l1', 'l2', 'elasticnet']}]
sgd_clf = GridSearchCV(SGDClassifier(random_state=22), parameters)
sgd_clf.fit(X_train, y_train)
```

Experimental Results: We saved 20% of the data for testing. Now we can compare the models by seeing their performance on this data. We can look at their F1-scores, and to help us visualize their performance, their confusion matrices.

Precision: Precision indicates what proportion of predicted positive reviews is truly positive.

Recall: Recall represents what proportion of actual positive reviews are correctly classified by the classifier.

F1-Score: Comparing two models with low accuracy and high recall is complicated, or vice versa. Therefore, the F1-Score is used to make them analogous. F1-score is known as a harmonic mean of Recall and Precision.

Accuracy of the models are recorded in the Figure 8.

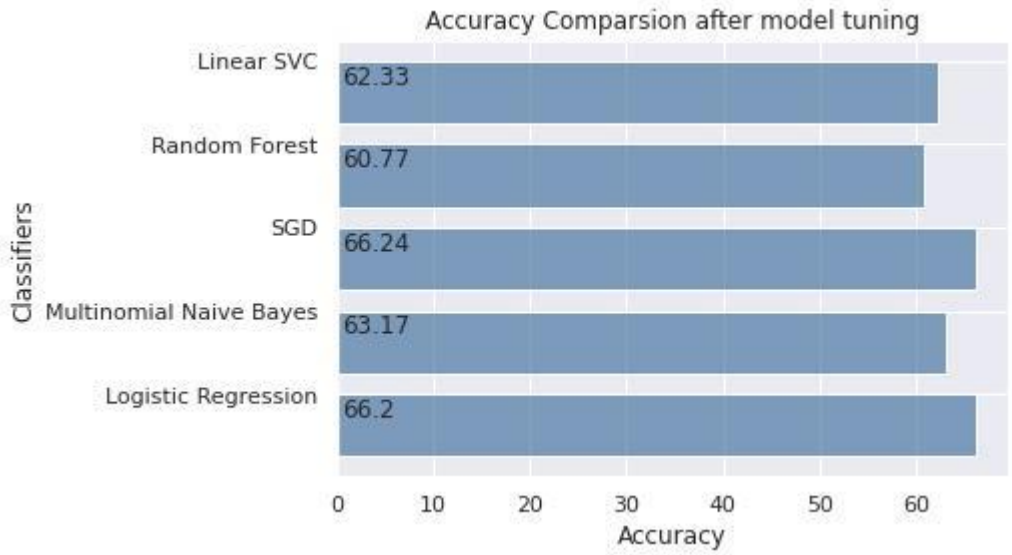
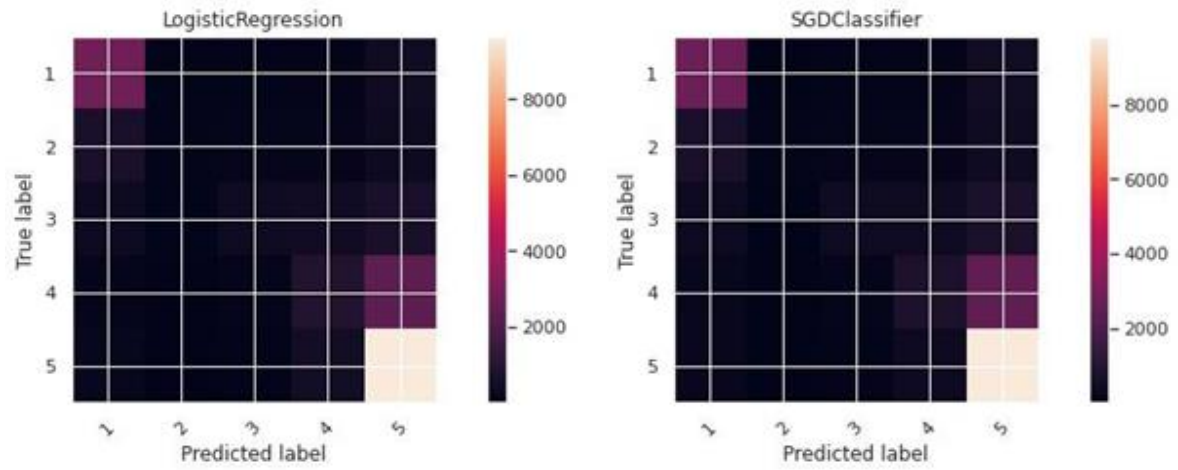


Figure 8: Accuracy Comparison

Confusion matrix is an important metric used to measure classifier efficiency. It represents the number of samples correctly classified and is used to determine precision, recall and F1 score. Confusion matrices are shown below. It is clearly observable that all the classifiers predict 1-star and 5-star reviews with much higher accuracy than the neutral reviews (3-star).



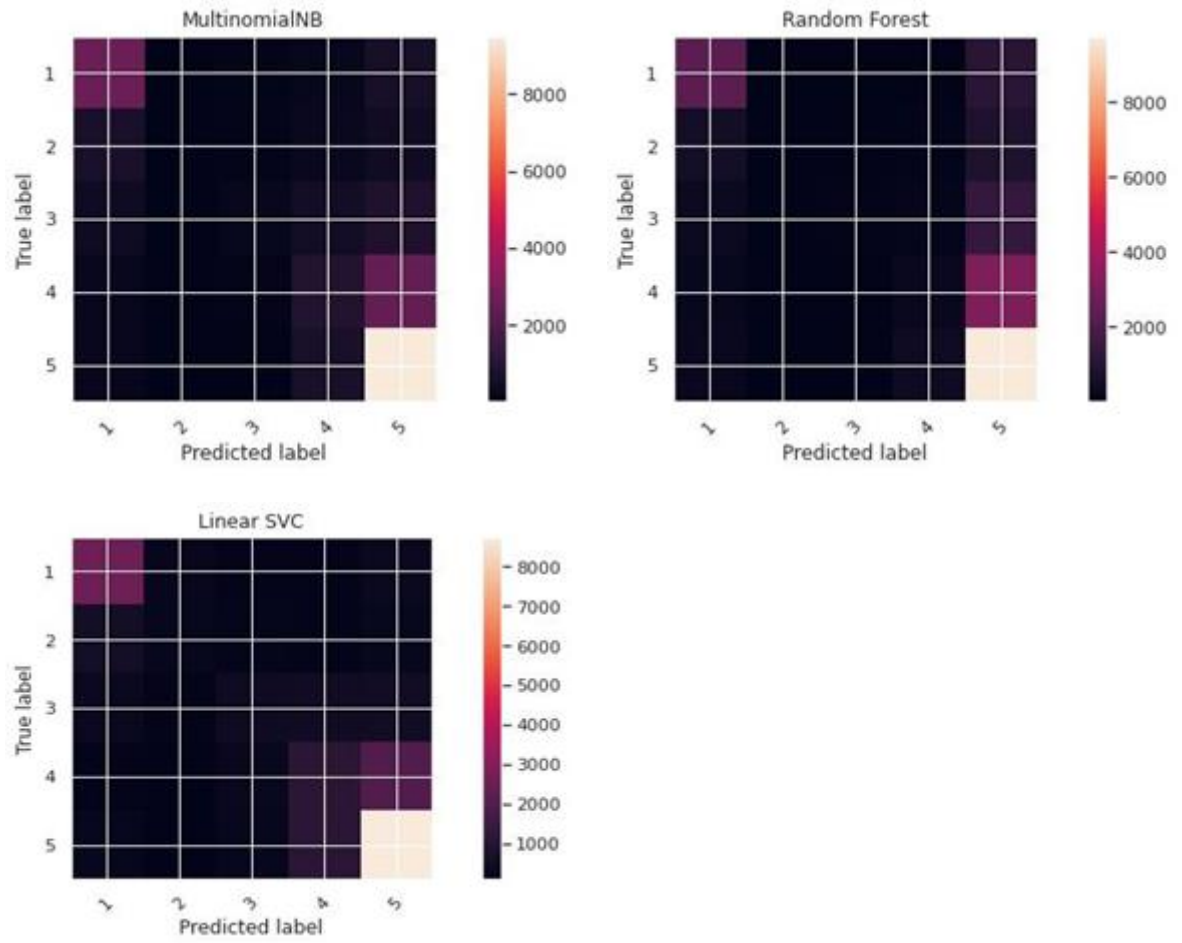


Figure 9: Confusion matrices

As we have five rating classes, any random guessing would be correct only 20% of the time which is our baseline. These confusion matrices tell us that the models have an easier time with the 5-star and 1-star reviews than the other reviews. Reviews with 5-star and 1-star are probably more straightforward because people who write these reviews use very distinct words.

Conclusion and Future Work: The mobile electronics dataset of Amazon.com was used to get some insights of the textual reviews of the products. We tried to understand if star rating classification can be done based on the review text and whether sentiments in the review text match with the star rating or not. We observed that a model can be developed which can

classify star rating based on the review description but has bit lower accuracy. The ground reality was analyzed using TextBlob which identifies the reviews showing a discrepancy with the user-provided ratings. Results demonstrate that the model of Logistic regression performs better than the rest of the classifiers. Even for a human, it's difficult to differentiate between these reviews. The difference between a 3-star and 4-star review is very subtle. Moreover, it can also be observed and concluded that it is relatively easy to predict 1-star and 5-star ratings owing to the presence of strong sentiments in them than in neutral reviews with 3-star ratings. The analysis also reveals that the user reviews are inconsistent with user numeric ratings. An area to look for improvements are ways to improve recall for these mid-star reviews. A solution not explored in this report is a neural network. Many people have found success using neural networks for text classification problems. A good candidate would be the use of the Keras Python library that would be useful in building a neural network model to predict ratings.

References

(2015, June 07). Retrieved November 28, 2020, from https://planspace.org/20150607-textblob_sentiment/

(n.d.). Retrieved November 26, 2020, from https://s3.amazonaws.com/amazon-reviews-pds/readme.html

3.2.4.1.5. sklearn.linear_model.LogisticRegressionCV. (n.d.). Retrieved November 28, 2020, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegressionCV.html

Band, A. (2020, September 27). Text2emotion: Python package to detect emotions from textual data. Retrieved November 28, 2020, from https://towardsdatascience.com/text2emotion-python-package-to-detect-emotions-from-textual-data-b2e7b7ce1153

Carpenter, J. (2020, June 04). Swiftapply - automatically efficient pandas apply operations. Retrieved November 28, 2020, from https://medium.com/@jmcarpenter2/swiftapply-automatically-efficient-pandas-apply-operations-50e1058909f9

Customer Sentiment Analysis: A crucial need in E-Commerce Data Initiatives |. (2020, August 31). Retrieved November 20, 2020, from https://www.nabler.com/articles/customer-sentiment-analysis-a-crucial-need-in-e-commerce-data-initiatives/

Jones, A. (2018, March 15). Sentiment analysis of reviews: Text Pre-processing. Retrieved November 28, 2020, from https://medium.com/@annabiancajones/sentiment-analysis-of-reviews-text-pre-processing-6359343784fb

Joshi, P. (2020, July 19). A NLP Approach to Mining Online Reviews using Topic Modeling. Retrieved November 26, 2020, from https://www.analyticsvidhya.com/blog/2018/10/mining-online-reviews-topic-modeling-lda/

Joshi, P. (2020, July 19). A NLP Approach to Mining Online Reviews using Topic Modeling. Retrieved November 28, 2020, from https://www.analyticsvidhya.com/blog/2018/10/mining-online-reviews-topic-modeling-lda/

Menzli, A. (2020, January 28). My NLP learning journey. Retrieved November 28, 2020, from https://towardsdatascience.com/my-nlp-learning-journey-6de23cf2b196

Mining Online Reviews using Topic Modeling with NLP. (2019, March 25). Retrieved November 28, 2020, from https://medium.com/@studyammu/mining-online-reviews-using-topic-modeling-with-nlp-d352ed21c5ef

Mukherjee, A. (2020, June 14). Online Rating Systems using Bayesian Adjusted Ratings. Retrieved November 26, 2020, from https://www.analyticsvidhya.com/blog/2019/07/introduction-online-rating-systems-bayesian-adjusted-rating/

Rai, A. (2019, January 23). Python: Sentiment Analysis using VADER. Retrieved November 28, 2020, from https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/

Raj, R. (2020, October 21). What is Random Forest and why is it so effective? Retrieved November 28, 2020, from https://medium.com/analytics-vidhya/what-is-random-forest-and-why-is-it-so-effective-3883c29b2b67

Saxena, M. (2018, July 06). Multi-Class Text Classification with Probability Prediction for each Class using LinearSVC in... Retrieved November 28, 2020, from https://medium.com/@manoveg/multi-class-text-classification-with-probability-prediction-for-each-class-using-linearsvc-in-289189fbb100

Umer, M., Ashraf, I., Mehmood, A., Ullah, S., & Choi, G. (2020, July 06). Predicting numeric ratings for Google apps using text features and ensemble learning. Retrieved November 26, 2020, from https://onlinelibrary.wiley.com/doi/full/10.4218/etrij.2019-0443

Ware, C., Aguglia, B., & Verma, U. Natural Language Processing (NLP) Using Python. Retrieved November 26, from https://courses.analyticsvidhya.com/courses/natural-language-processing-nlp?utm_source=home_blog_navbar