Data Mining Final Report

-- Data Predictive Analytics & Text Mining

on Amazon Food Reviews Data

Submitted To:

Professor Artur Dubrawski

Submitted By:

Team A

Bhavya Chilakapati

Liduvina Cisneros

Muhammad Ali Asad Khan

Xinyu Wang

Enbo Zhang

Submitted On:

February 4, 2015

Problem Statement – Business Scenario And Major Drivers

The client has a dataset consisting of 568,454 reviews of various food items obtained from Amazon's website. There are various attributes related to identifying users, products, and time the review was made. Others relate to the score given to the product by the reviewer, a summary of the review, and the text of the review itself. There is also an attribute on the "helpfulness" of the review, presumably to other users on the website.

Based on our understanding of the requirements in the project document and our interaction with the client, the specific goals and client needs for this engagement are:

- ***Create Insights From The Data:*** The client wishes to extract value from the data. They want answers to questions such as what is a valuable product? They would like to see metrics such as distributions of ratings for products, average ratings, and the number of users reviewing a given product to understand how valuable it is. If a product is given good reviews, then why is it good? What are the reasons (keywords) that can help make such a distinction? How can reviews be characterized? They would also like to extract complaints from the reviews that can improve Amazon's business processes. For instance, a product may be highly rated and given an excellent review, but there may be a complaint about late delivery or other process issues that impact Amazon negatively.
- ***Create Predictive Intelligence:*** A major goal of this project is to use the existing data and create a model that can predict scores based on the inputs. The client envisions scraping similar data from competitors' websites, blogs, etc. for food items which it does not yet currently offer and for which there are no scores. The model will be applied to this data and scores will be predicted – enabling the client to identify food products that consumers would likely rate highly. This predictive ability will be able to inform their decision on whether to offer those products on their own website.

Getting the Raw Data

The data was provided in the form of a flat file consisting of millions of lines, where the attributes were arranged in rows instead of columns and each observation was spread out over multiple lines. We extracted this raw data and arranged it all into a data frame object. The data was stored into a binary .Rdata file which be accessed by simply loading it into R.

Data Visualizing -- Good vs. Bad reviews

Scores and Helpfulness

First of all, we visualized the distribution of scores. According to Chart 1, we can see a severely skewed distribution of frequency of scores[1]. Score 4 and 5 take up more than 80% of all scores. For further analysis, the neutral score 3 was eliminated from our data

[1] The frequency here refers to the absolute number of times that each score appears. It's not a proportion.

set. After this process, Score 4 and 5 take up more than 90% of all scores, showing an even more inbalanced distribution.

Chart 1:

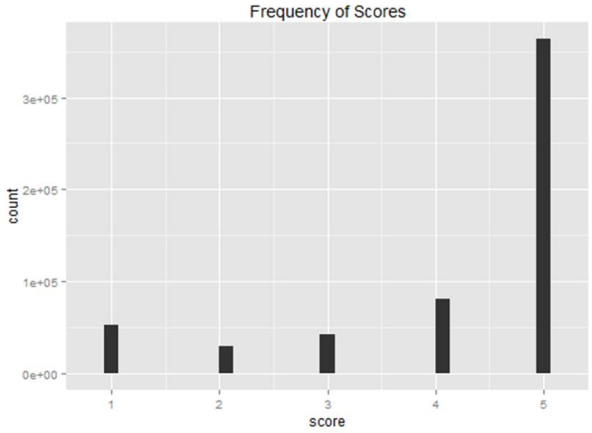
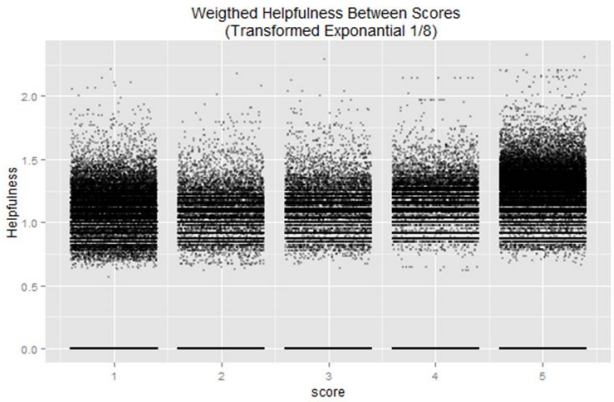


Chart 2



Helpfulness is another attribute that we focus on. Scores with higher helpfulness should be put on more weight. We used n^2/m as the y-value in the plot, n being the number of people thinking the review is helpful, and m being the total number of people voting on the review. We also transformed the scores into exponantial ⅛ in order to expand the denser reginos. The result shows some interesting patterns:

•The reviews with highest scores were deemed most helpful

•The reviews with the least scores were deemed the next useful

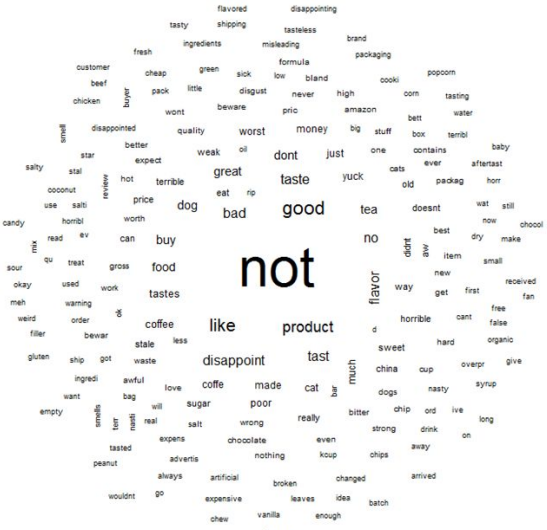
•This shows that users care most about extreme reviews

•Score 2 and 3 are the least useful

Word Cloud

We made two word clouds for reviews with low and high scores, to briefly check the frequency of words in each score class.[2]

word cloud for score 1 & 2



word cloud for score 4&5

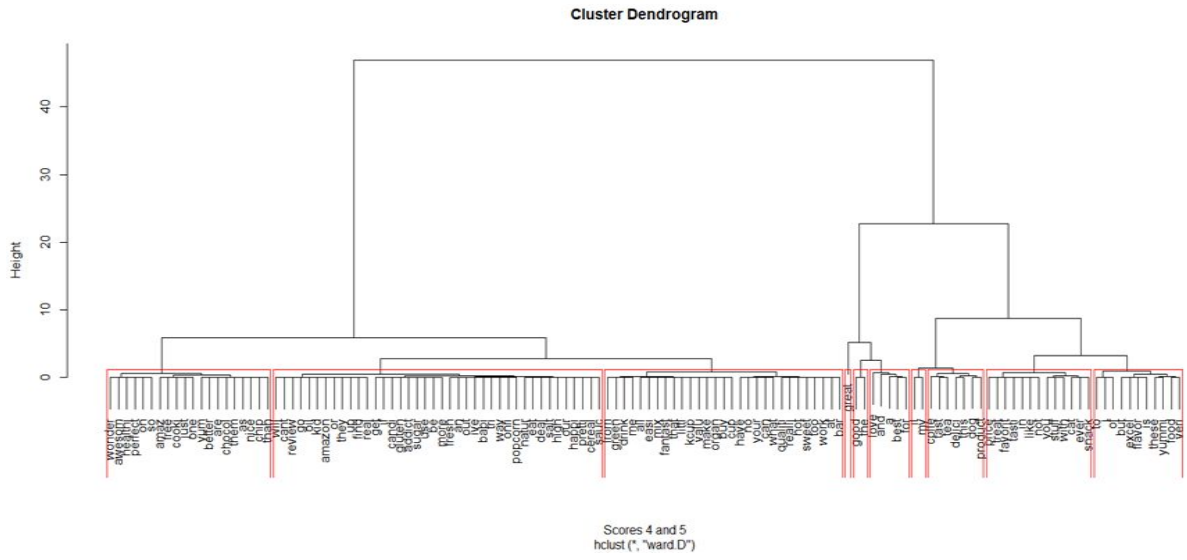
[2] Just to clarify, the word cloud is not based on the number of times each word appears in all reviews. Instead it's based on the number of reviews in which each word appears.

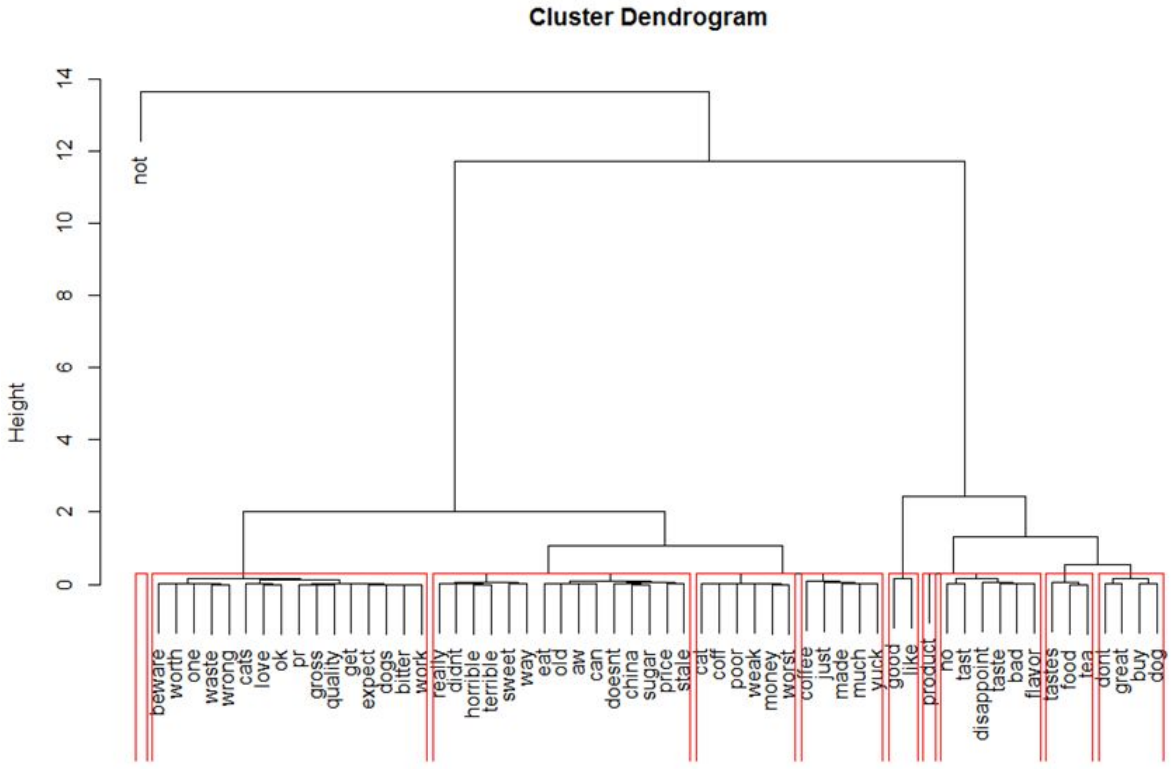
From the two images above, we find an interesting thing that positive words, like great, good, also appear in word cloud for low score. It is reasonable because it might be accompanied with negative word, say not. Our model could find inner correlation between words and solve this problem to some extent. However, if we expand our analysis scope to n-gram, we could get better result. For example, if we look into two-gram, we could find out that “don’t like” would appear frequently in low score word cloud whereas it won’t appear in high score word cloud. It can indicate more semantic information.

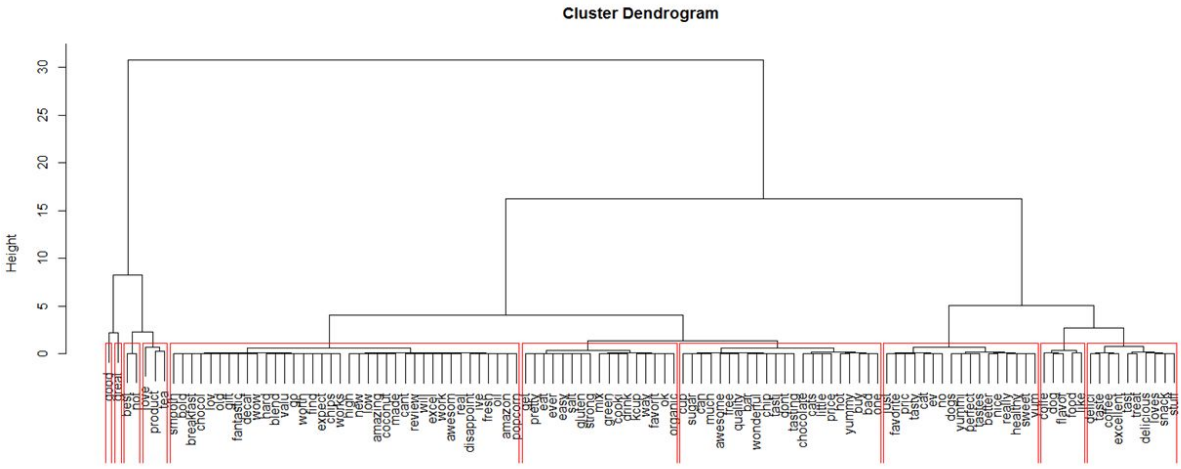
Clustering

We also performed clustering on the words. For interpretation purposes, the height denotes the relative scoring importance of one word with respect to another. For example, the word “NOT“ here occurs in every review irrespective of the presence of other words. Also, coming to the clades, words that are closely related are placed in related clades. For Ex: weak, worst, just, poor are all part of the one clade. Weak & worst are equally distant from horrible or terrible because each of them is in a cluster before joining the other set. Good & like are not particularly close to any of the other words since they join much later.

GOOD, BEST & LOVE are equally distant from GREAT. These are much closely related before any other cluster of words join them and they join other clusters over a height of 20 meaning they are least closely related to any other clusters. Interestingly on combining the good and bad reviews, the words best and not seem to be very closely related in this cluster, suggesting a “not the best….” pattern.







Data Sampling & Preprocessing

After a brief look, the raw data was sampled and preprocessing was implemented for model training.

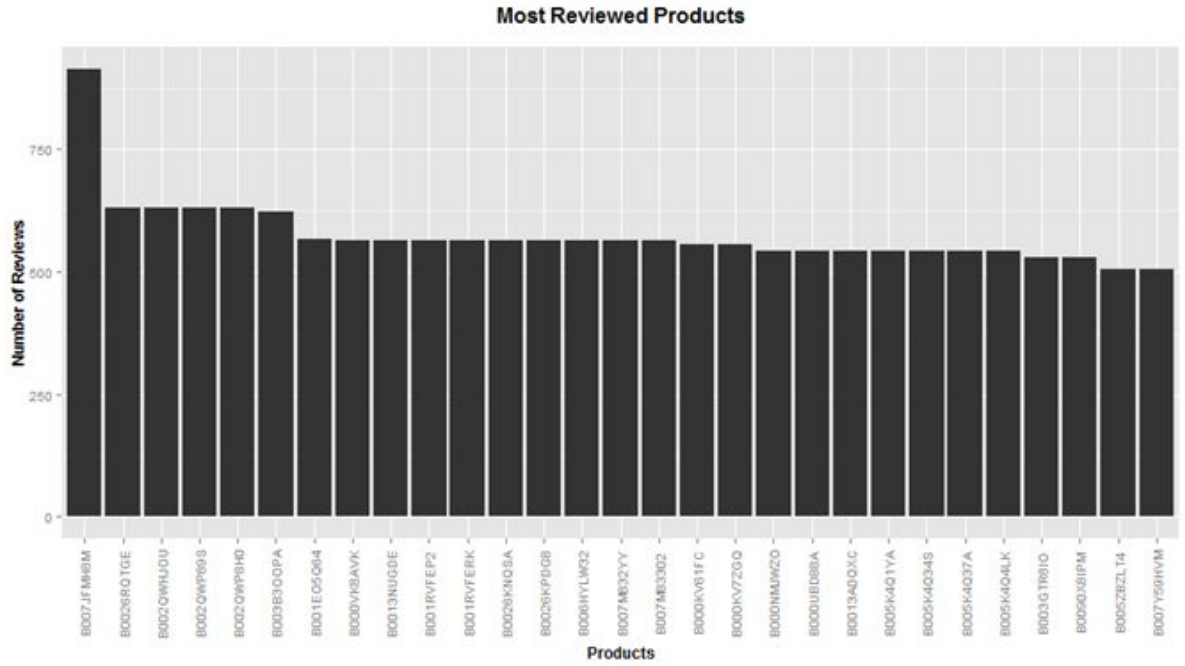
To help our model better learn the data, we need a balanced data set. We performed stratified sampling, randomly taking out 5,000 observations with low scores and 5,000 with high scores, together forming a sample with the length of 10,000. We have to train the models with sampled data because otherwise we would face memory issues and lengthy run-times.

Two corpuses were created, one each for the summary and review attributes. On each corpus, the following operations were performed:

- Replacing '/', '@', '|' which is sometimes used to seperate alternative words with a space
- Converting to lower case
- Removing numbers
- Removing punctuation
- Removing english stop words
- Stripping extra white space
- Stemming

After these basic cleanup operations, two document term matrices (DTM) were created, one each for the summary and review attributes. The DTMs were weighted using TfIDf. Sparse terms were removed at a level of 95%.

We visualized the preprocessed DTMs to find some patterns in them. We only took into account products reviewed no less than 500 times. The result, we believe, has business impact in that Amazon can explore its most reviewed products, analyze what people say about them, cross-tabulate them with sales.



The next thing we did was processing the "helpfulness" attribute. We separated the numerator and the denominator in that attribute. We expressed a new attribute as helpPercent, which is just the percentage of people who found it helpful. We also developed a new attribute helpWeight, which is just the denominator, i.e. the total

number of people who found the review helpful (in order to weight the percentage, so that reviews found helpful by a higher number of people are weighted more).

For the purposes of this analysis, we excluded “profileName”, “time”, and of course “help”, which had already been featurized into two new attributes. After that, the DTMs created earlier were joint to the data we had featurized, and ready for further modeling.

Modeling

Generalization

We are going to set up the problem as a binary problem by dividing scroes from 0~5 into two general categories: one is good score set and another one is bad score set. Based on previous analyse, we already knew that score 3 is a neutral score and it could be served as the threshold between good scores and bad scores. That’s to say, we regard score of 1~3 as bad products and that of 4~5 as good products. In reality, Amazon(client) might specify the base line and thus it is much more accurate to discriminate types of goods.

Data Pre-process

- Entire is shuffled to remove potential causal order.
- “ProductID” and “UserID” are removed from attributes when the dataset is being process as these two columns have too many factor levels, which nevertheless is not help to our model.

Basic Approach

- Take 10,000 records for modeling to get distribution of scores preserved.
- Steps repeated for samples of 10,000 with a 50-50 distribution of good and bad scores (to avoid bias in original models; biased towards predicting good scores/products due to 80-20 distribution of scores in favour of good.)
- In order to find the best classification model and its parameter, we use 10-fold cross validation to train the following four models: Logistic Regression, Support Vector Machines (kernel: radial basis), Naïve Bayes and KNN (k varied from 5-25, ignoring 1- 4 in case of over-fitting).
- Run it two times on different samples of 10,000 each and compile results.
- Compile results and find out the best model and its parameter if needed.

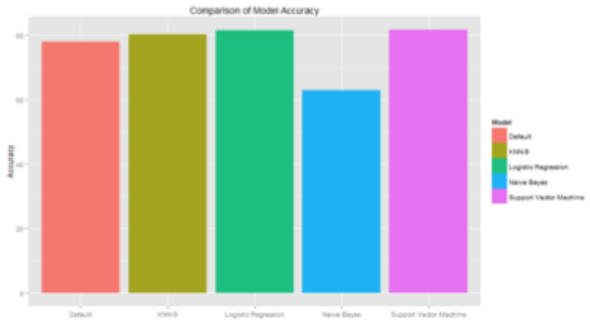
Model Selection

We compare four models mentioned above regarding to ther accuracies, ROC curves and list the result as follow.

1. Input with original 80-20 distribution of score preserved.

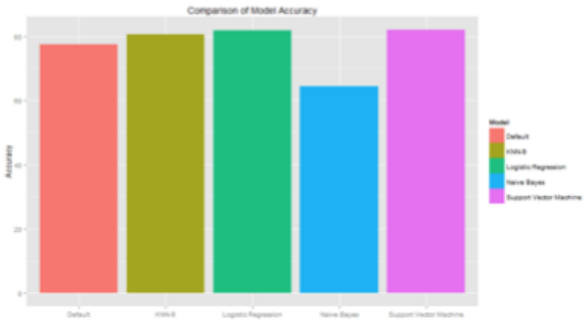
Comparison Of Model Accuracies

FIRST RUN



NOTE: This is KNN-17. Not KNN-9.

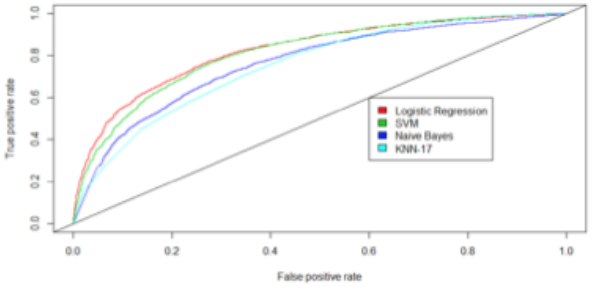
SECOND RUN



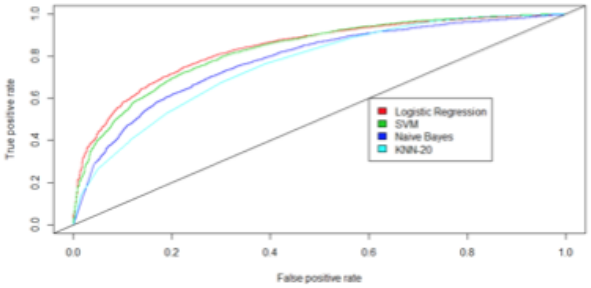
NOTE: This is KNN-20. Not KNN-9.

ROC Curves

FIRST RUN



SECOND RUN



Performance Metrics

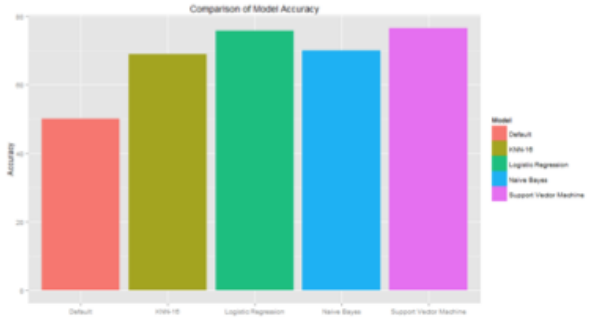
Model	Area Under The Curve (AUC)	Maximum Accuracy
Logisitic Regression	0.825	0.815
SVM	0.815	0.816
Naïve Bayes	0.764	0.793
KNN-17	0.754	0.807

Model	Area Under The Curve (AUC)	Maximum Accuracy
Logisitic Regression	0.838	0.819
SVM	0.829	0.823
Naïve Bayes	0.781	0.798
KNN-20	0.760	0.811

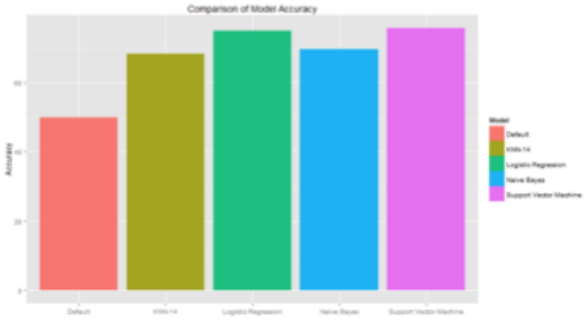
2. Input with original 50-50 distribution of score preserved.

Comparison Of Model Accuracies

FIRST RUN

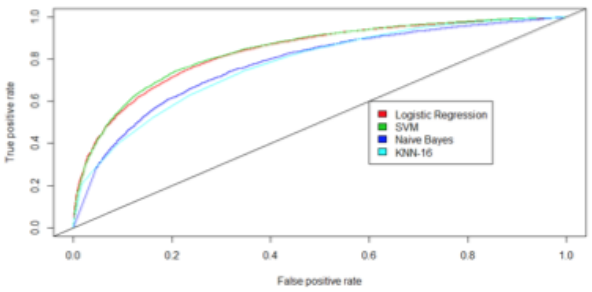


SECOND RUN

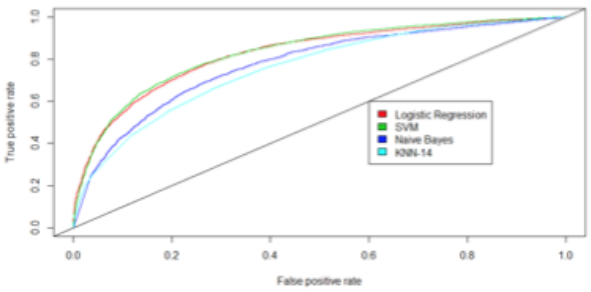


ROC Curves

FIRST RUN



SECOND RUN



Performance Metrics

Model	Area Under The Curve (AUC)	Maximum Accuracy
Logisitic Regression	0.837	0.760
SVM	0.841	0.769
Naïve Bayes	0.777	0.713
KNN-16	0.774	0.699

Model	Area Under The Curve (AUC)	Maximum Accuracy
Logisitic Regression	0.828	0.754
SVM	0.833	0.759
Naïve Bayes	0.776	0.712
KNN-14	0.759	0.688

Judging from above graphs and tables, we can draw conclusions that:

- Taken all into consideration, SVM performs best.
- 80-20 distribution of scores in favor of good preserves more information of original data.
- KNN might face over-fitting problem.
- Logestic regression might be good to use when false positive rate is low.
- To be added

Business Impact

- **Corporate Knowledge:**
  For products of a certain brand, this text mining enables Amazon to help the brand maintain an overview of what it knows about a certain topic. For example: is a certain major ingredient replaceable?

- **Market Intelligence:**
  Text Mining can help Amazon identify potential customers across geographically diverse markets or across diverse age groups. By deciding search parameters based on current reviews, other websites can be targeted for gathering further information to build a list of prospective customers.
- **Insights:**
  These analyses, with more work, will potentially help Amazon understand what is working in favor of their competitors and also what is the expectation of the general market in terms of likeability of a product.

Future Work

Fine Tuning Existing Models

SVM model could perform better if we can explore different parameters and kernel function settings to make.

Multi Dimensional Scaling

Develope n-grams taking more successive words into account and thus it can preserve more information.

More Nuanced Sentiment Analysis.

- Custom lexicon, dictionaries, and stop lists.
- Using ngrams as inputs, e.g. word pairs.
- Would help in preserving semantic structure/sentiment, especially for negatives, such as “don’t like”.

Principal Components Analysis

Try to use PCA before remove sparse terms. PCA can analyze word appearance of all review text and extract the most significant words. We could get more words involed as we set parameter correctly than what we get from TfIDf.

Other Models

- Decision Trees and Random Forests (Bootstrapping and Bagging)
- Step-wise Regression (attribute selection)
- Neural Networks.
- LambdaMARTS (Ranking algorithm)
- Elastic Nets/Bayesian Shrinkage.