# Product Category Prediction Based On Amazon Customer Reviews

# Introduction to Data Science Project Spring 2017

Professor Arthur Spirling | Kevin Munger (TA)

# New York University

# **Arun Govindaiah Sharang Biradar Danni Lu**

**ag5305@nyu.edu sdb418@nyu.edu dl3253@nyu.edu**

**Business Understanding**

Marketers are able to be more precise with their advertising today than ever before, supported by more and better data, and evolving technology and tool. More precision means less waste, less advertising directed at people who won't respond. Less waste means a better return on investment. As the ROI of advertising improves, brands allocate a greater portion of their total marketing spend, and a greater portion of their brand's revenues, to advertising.

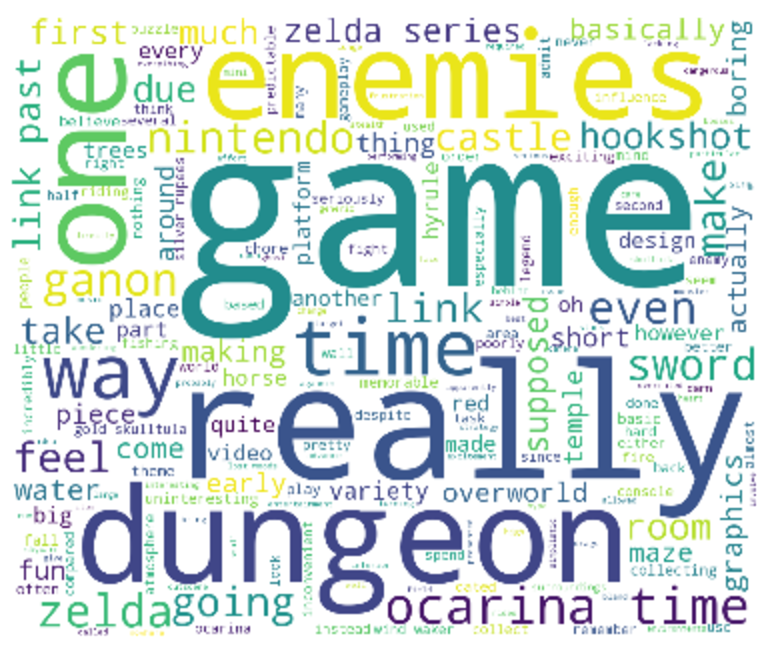
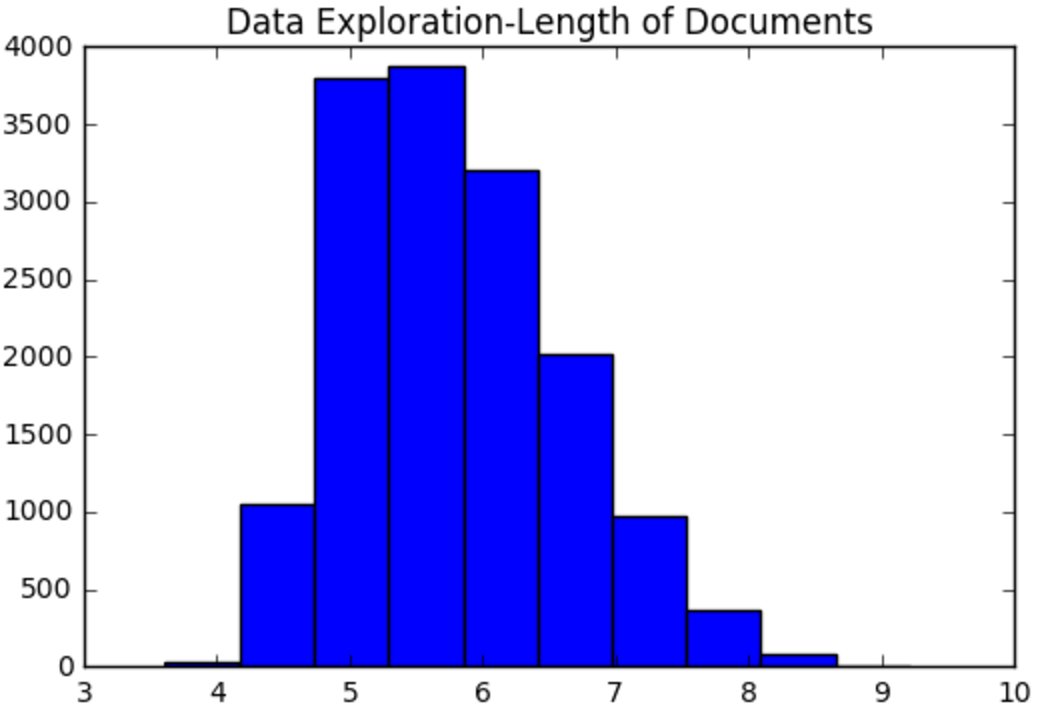
In this pursuit, our project aims to build a multiclass classifier to predict a product category that a given textual data snippet is talking about. Using such a prediction allows advertising agencies to target their audience with the promotional content with more precision and relevance. Manual prediction and analysis is simply impossible with the sheer amount of data and so it warrants a data science solution.

**Data Understanding**

As one of the world’s largest online retailers, Amazon.com has a sales volume of tens of billions dollars and as a result huge amounts of customer reviews accumulate. These reviews are great learning samples for our prediction model. Amazon’s user review data has 24 product categories out of which, 22 have been chosen eliminating 2 overlapping categories. Out of millions of reviews, a 1000 each have been selected per category making a total of 22000 reviews. This dataset is made available by Stanford University’s **S**tanford **N**etwork **A**nalysis **P**latform (**SNAP**) and Julian McAuley (UCSD).

**Data Preparation**

From millions of reviews for each category, we have chosen reviews that are at least 100 characters each so we have enough data for comprehensive training and testing. The original dataset has product ID, price, title, category, related products, reviewer name, summary, review text, etc. In this project, we are using only the review text and its associated product category for classifying. We are trying to predict the product category (22 categories including Musical Instruments, Baby, Beauty, Apps for Android, Digital Music, Office Products, Automobile, Pet Supplies, Grocery, Movies & TV, Toys And Games, Patio Lawn Garden, Tools & Home Improvement, CD & Vinyl , Sports And Outdoors, Electronics, Music Instruments, Amazon Instant Video, Cellphone Accessories, Health, Clothing & Jewellery, Home And Kitchen, Videogames, Books) which is the target variable. The review text is converted to lowercase and special characters are removed from the data.



*Figure: Number of documents VS log(length of doc), Word Frequencies in “Toys And Games”*

NLTK corpus is used for removing stop words from the dataset. Additional stop words were added to the stop word list based on the data manually. The remaining words undergo further processing like

Tokens are generated from 22000 reviews. With the bag-of-words model, a dictionary of all the 43886 tokens constructed and each review is transformed into a Boolean-valued 1×43886 vector. (An element is labeled 1 if its corresponding token appears in the review and 0 otherwise.) and a counts-vector (An element is labeled the number of times its corresponding token appears in the review). The dataset is randomly split into training and test data in the ratio 70:30.

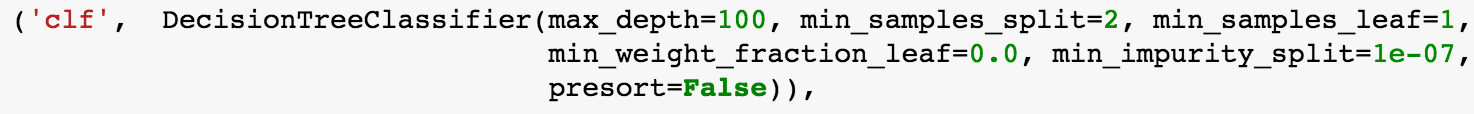
**Modeling and Evaluation**

We have built 5 multiclass classifiers, namely,

* Decision Tree Classifier
* Stochastic Gradient Descent Classifier
* Multinomial Naive Bayes Classifier
* Random Forest Classifier
* Logistic Regression

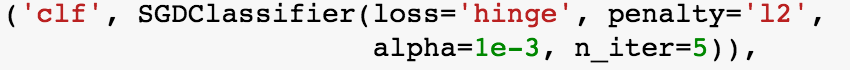
We will compare the performance of each of these classifiers and discussed the results. A suitable model will be identified to solve the problem and steps to deploy the same will be summarized. The five classifiers used are discussed briefly below.

**Decision Trees Classifier** is a non-parametric supervised learning method used for classification and regression. A decision tree classifies data items by posing a series of questions about the features associated with the items. Each question is contained in a node, and every internal node points to one child node for each possible answer to its question. The questions thereby form a hierarchy, encoded as a tree. In the simplest form, we ask yes-or-no questions, and each internal node has a ‘yes’ child and a ‘no’ child. An item is sorted into a class by following the path from the topmost node, the root, to a node without children, a leaf, per the answers that apply to the item under consideration. An item is assigned to the class that has been associated with the leaf it reaches. In some variations, each leaf contains a probability distribution over the classes that estimates the conditional probability that an item reaching the leaf belongs to a given class.



**SGD Classifier** Linear Classifiers (SVM used here) with SGD (Stochastic Gradient Descent) training. This estimator implements regularized linear model with stochastic gradient descent (SGD) learning, the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule (aka learning rate). This implementation works with data represented as dense or sparse arrays of floating point values for the features. The model it fits can be controlled with the loss parameter and it fits a linear support vector machine (SVM).

The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero vector using either the squared euclidean norm L2 or the absolute norm L1 or a combination of both (Elastic Net). If the parameter update crosses the 0.0 value because of the regularizer, the update is truncated to 0.0 to allow for learning sparse models and achieve feature selection.



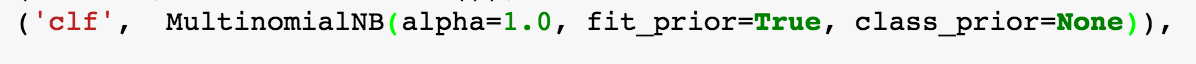
**MultinomialNB Classifier** implements the Naive Bayes algorithm for multinomial distributed data, and is used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors \theta_y = (\theta_{y1},\ldots,\theta_{yn}) for each class y, where n is the number of features (in text classification, the size of the vocabulary) and \theta_{yi} is the probability P(x_i \mid y) of feature i appearing in a sample belonging to class y.

The parameters \theta_y is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:

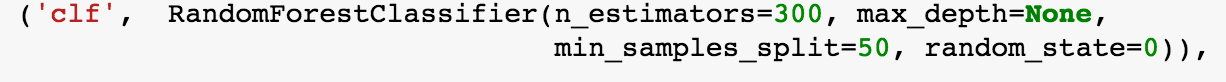
\hat{\theta}_{yi} = \frac{ N_{yi} + \alpha}{N_y + \alpha n}

where N_{yi} = \sum_{x \in T} x_i is the number of times feature i appears in a sample of class y in the training set T, and N_{y} = \sum_{i=1}^{|T|} N_{yi} is the total count of all features for class y.

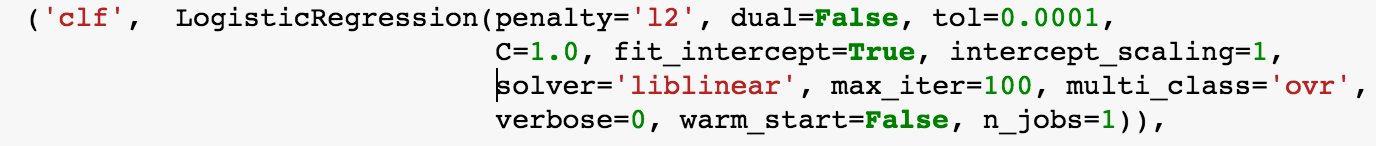
The smoothing priors \alpha \ge 0 accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting \alpha = 1 is called Laplace smoothing, while \alpha < 1 is called Lidstone smoothing.



**Random forests** or random decision forestsare 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 (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. In random forests each tree in the ensemble is built from a sample drawn with replacement from the training set. In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model.



**Logistic regression**, or **logit regression**, or **logit model**is a regression model where the dependent variable (DV) is categorical. **multinomial logistic regression** is a classification method that generalizes logistic regression to multiclass problems, i.e. with more than two possible discrete outcomes.That is, it is a model that is used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables.



Multinomial Naive Bayes model with binary-valued tokens as features is a suitable candidate for our baseline model. The feature vector is a binary valued vector of size (1x43886). It performs fairly good with its accuracy at 84.27%. Here is a table with the performance of all the different classifiers for features with binary-valued tokens.

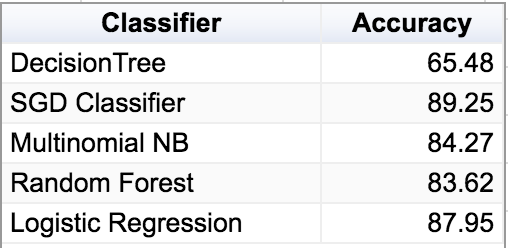


Figure: Model performances with binary-valued feature vector.

To improve performance, we tried to change tokenization techniques. The tokens are now counts based. The feature vector is not integer-valued of size (1X43886). Each element is an integer corresponding to the number of times the token appears in the document. The resultant performances of the models have been tabulated below.

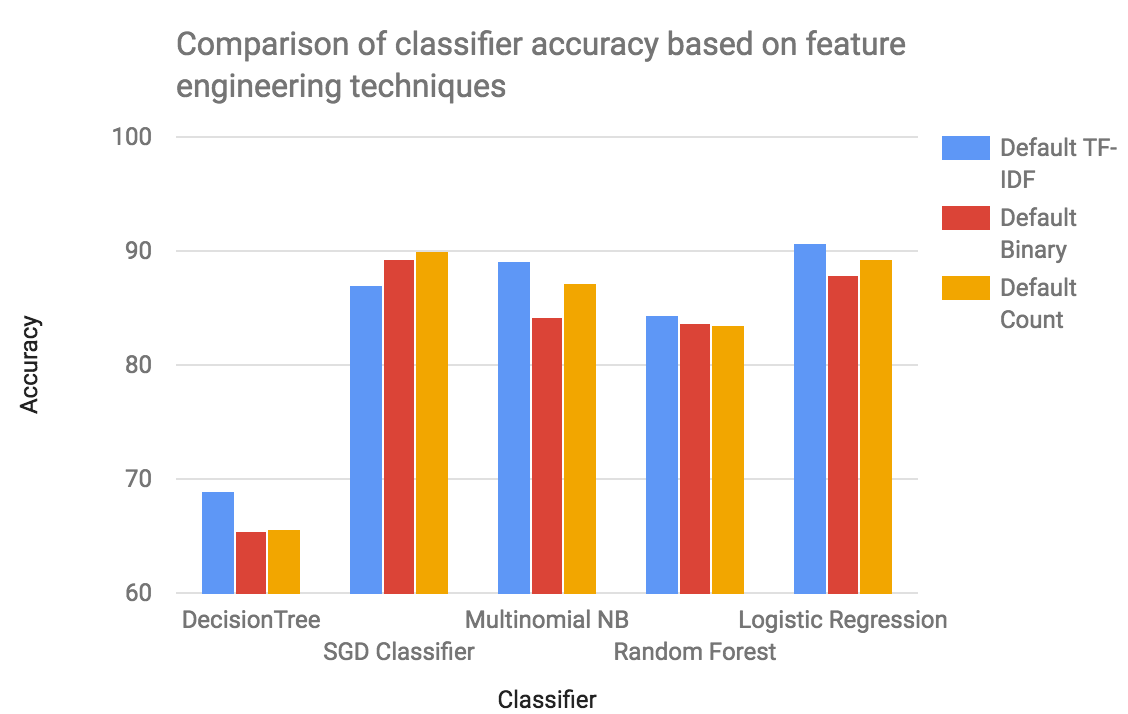


Figure: Model performances with counts feature vector

The performance improves greatly when we incorporate TF-IDF vectorizing and transforming. The results have been tabulated below.



Figure: Model performances with default TF-IDF



However, when we tested bi-grams and tri-grams the performance is drastically reduced. It shows that it is not so important to consider phrases (two or three words) when we are building a model for review text classification. We see this from the accuracies that are well below those for unigrams based models.

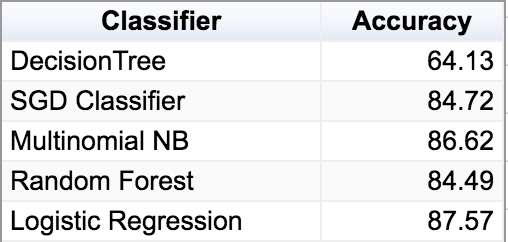
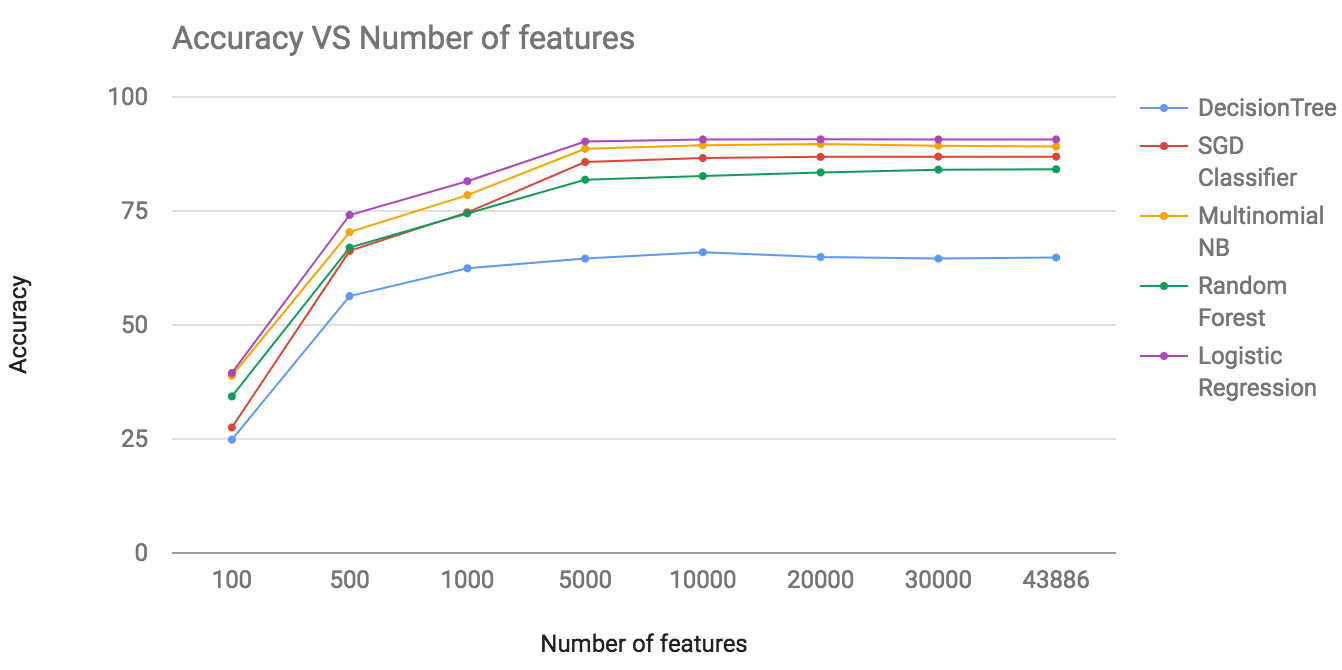
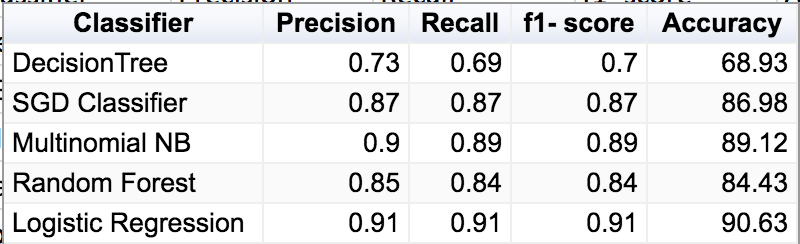


Figure: Classifier performance for bi-grams

Feature Selection: We sampled by change the number of features to test the performances of all the models. First, we chose randomly chose the features where the performances were dismal and then we picked the top features ordered by their term frequencies across the corpus. We have chosen the following numbers to have a non linear scale : 100, 500, 1000, 5000, 10000, 20000, 30000, 43886(max).



Analysis of the performances across different number of features reveals that all of the classifiers are performing better at around 20000 features (about roughly half of the total size) and remains more or less same through the total number of features. Overall performances of all the classifiers at 20000 are shown below.



We see the that Logistic regression performs better than all of the other classifiers at all sizes of the features. The peak accuracy of Logistic Regression is 90.63%. It is closely followed by Multinomial Naive Bayes with an accuracy of 89.12%. Decision tree performs the worst. The peak accuracy is 68.93% only.

**Discussion**

This model can be deployed by advertising agencies as a program that automatically predicts the product category a user is talking about in various social media platforms like Twitter. This can also be used in browsers and search engines and places where a user would potentially enter information about a product. The relevant ads can be targeted based on the product category predicted. We have used Amazon’s dataset. However, the model can be updated regularly with datasets from other sources such as Yelp and eBay.

This is a model based on textual data entered by the user, however, ad agencies and organization can improve the model further by incorporating other user attributes such as browsing history and demographic information.

Ethical considerations in this model is along the same lines of overall machine learning. There are privacy concerns associated with the use of private data and what should be private and what should be not. Also, there are questions like “Would you allow a machine to make a decision on your behalf? Having mentioned these concerns, one should note that these are not products of our model but are inherent in machine learning itself.

Some of the risks associated with our proposed plan are the problems of classifiers over time. Same model cannot be used continuously without updating. And also, learning with more and more data as time passes may not be a good idea. We should be wary of associated issues like context shifts, survivorship bias and selection biases.

**Team Contribution**

We all contributed equally to this project, with most of the work done through pair programming in group meetings.

Arun - 33.33% - Worked on bag of words featurization, applying various machine learning classifiers, feature selection, parameter tuning, generating visualizations, report write-up.

Sharang - 33.33% - Worked on initial data analysis, initial TF-IDF/bag of words features, machine learning classifiers, data exploration with word cloud, visualization of results, report write-up.

Danni - 33.33% - Worked on initial data exploration, visualizations of results, feature engineering, ethical considerations, analysis, report write-up.

**Acknowledgment**

We would like to thank Professor Arthur Spirling and Kevin Munger who have provided us with so many suggestion. We have learnt a lot from our regular meetings every Thursday. Also, we would like to thank the Stanford SNAP portal and Julian McAuley (UCSD) for providing us with the data.

**References**

1. Kamal Nigam, Andre Kachites Mccallum, Sebastian Thrun, Tom Mitchell, *“Text Classification from Labeled and Unlabeled Documents using EM”, Carnegie Mellon University, 1998*
2. Brank J., Grobelnik M., Milic-Frayling N., Mladenic D.*, “Interaction of Feature Selection Methods and Linear Classification Models”, Proc. of the 19th International Conference on Machine Learning, Australia, 2002.*
3. T-Joachims “*Text categorization with support vector machines: Learning with many relevant features”, Springer, 1998*
4. Abhiteja Gajjala, “*Multi Faceted Text Classification using Supervised Machine Learning Models”, San Jose University, 2016*
5. Siva Gangireddy, *“Supervised Learning for Multi-Domain Text Classification”, San Jose State University, 2016*