# Applying Machine Learning to Amazon Reviews

Kiran Dsouza
Dartmouth Colege

Abstract

This report describes a machine learning approach to match a product review with a rating that is broadly consistent with the reviewer's own rating. I analyze and categorize Amazon Reviews using a training dataset and evaluating the models learned on a test dataset. I encoded the datasets, applied four classification techniques available in Scikit Learn, identified the promising variants and used these to make predictions on the test dataset. All classification results were uploaded to Kaggle for evaluation and exceeded the baseline Macro F1 scores. My product category clustering analysis also exceeded the baseline Silhouette score..

## Introduction

When a shopper at Amazon writes a review, the final step of assigning a rating to the product is very subjective and may not match very consistently with what the review is actually saying. This is motivation to explore if there is an acceptable automatic way to generate the final rating which broadly matches the rating that shoppers are actually assigning with their review. Machine learning is one way to achieve this. The goal of this project is to analyze and categorize Amazon

product reviews using two datasets, one for training and one for test and assess how well the predicted rating matches the final user rating. I also do some clustering analysis to determine if the rating system from 1 to 5 is sufficient to capture the various classes of reviews.

After encoding the data, which was itself a major task, there are 3 overall methods that I applied to the datasets:

1. Binary classification
2. Multiclass Classification
3. Clustering

For classification I evaluated 4 techniques available in Scikit-learn[1] on the training set to determine the best one to model on the test set. For clustering I used just one method, K-means. The results of the model on the test set were uploaded to the Kaggle website for evaluation. All results exceeded the baseline macro F1 scores as shown in the following table:

| Task | Baseline Macro F1 Score | Macro F1 Score from Kaggle |
|---|---|---|
| Binary Classification Cutoff 1 | 0.70 | 0.71415 |
| Binary Classification Cutoff 2 | 0.78 | 0.81883 |
| Binary Classification Cutoff 3 | 0.80 | 0.83650 |
| Binary Classification Cutoff 4 | 0.70 | 0.81663 |
| Multiclass Classification | 0.47 | 0.54500 |

Furthermore, my clustering silhouette score of 0.63 also surpassed the baseline Silhouette score of 0.59.

## Related Work

There has been much work on sentiment analysis. Salmony[3] considers Amazon Reviews in a survey of sentiment analysis that covers both supervised and unsupervised learning like I did here. A related field is that of automated customer satisfaction. In [4], they apply similar machine learning techniques to ours, but they use NLTK which we avoided here.

## Methods

### *Encoding the features in the dataset.*

Most of the features in the Amazon review dataset were non-numeric. Furthermore, some of the features were textual, such as the review summary and the review text. For the simple categorical data, I assigned numeric identifiers from 0 to the number of unique values using the Sciikit-learn LabelEncoder. I am also evaluating using one-hot encoding. For the textual data, I used the vector encoding in Scikit learn, and based on experiments, picked the TfidfVectorEncoding. For missing values, I assigned 0 by default for numerical data, and the empty string otherwise. There is useful advice on data cleanup from John Foreman[2], though he does his analysis entirely in Excel.

***Classification Techniques and Hyperparameters***

The techniques used are shown in the table below. For various combinations of the hyperparameters, we performed 5-fold cross validation in Scikit-learn and computed the average of the 5 scores. This information was used to identify promising technique and hyperparameter combinations.

| **Technique** | **Hyperparameters** |
|---|---|
| MultinomialNB | fit_prior, alpha = 1, 0.5, 0.125, 0.0375, 0.01, 0.001, 0.00001,0 |
| LogisticRegression | solver |
| RandomForest | N_estimators = 25,50,75,100,125,150,175 |
| Adaboost | N_estimators =25, 50,75,100,125,150,175 |

## Results and Analysis

The following table shows the results of 5-fold cross validation for hyperparameter tuning for Multinomial Naive Bayers. The hyperparameter alpha controlled the amount of smoothing, with value 0 used for no smoothing. Generally small values of alpha worked better in all cases. Multinomial Naive Bayes was competitive with the other methods for Cutoffs 1, 3 and 4. It also made the baseline macro F1 score on Kaggle for Cutoff 1 and Cutoff 4.

| **Classifier used in 5-fold cross-validation** | **Cutoff 1 mean** | **Cutoff 2 mean** | **Cutoff 3 mean** | **Cutoff 4 mean** | **Multiclass mean** |
|---|---|---|---|---|---|
| MultinomialNB: | 0.819453213 | 0.767687179 | 0.80910704 | 0.872314768 | 0.522321 |
| MultinomialNB_no_smoothing: | 0.783618466 | 0.769023564 | 0.791257705 | 0.787487636 | 0.52191 |
| MultinomialNB_alpha_1: | 0.819453213 | 0.767687179 | 0.80910704 | 0.872314768 | 0.522321 |
| MultinomialNB_alpha_0.5: | 0.823050418 | 0.768714899 | 0.808661676 | 0.870670311 | 0.524548 |
| MultinomialNB_alpha_0.25: | 0.824934678 | 0.769674149 | 0.808353346 | 0.869985145 | 0.526021 |
| MultinomialNB_alpha_0.125: | 0.825414306 | 0.770016732 | 0.808147784 | 0.869711066 | 0.526192 |
| MultinomialNB_alpha_0.0625: | 0.825791147 | 0.770290828 | 0.807942228 | 0.869539781 | 0.526843 |
| MultinomialNB_alpha_0.03125: | 0.826133724 | 0.770325087 | 0.807873712 | 0.869676826 | 0.526603 |
| MultinomialNB_alpha_0.01: | 0.826167977 | 0.770633411 | 0.807805178 | 0.869745348 | 0.526569 |
| MultinomialNB_alpha_0.001: | 0.826236517 | 0.77046212 | 0.807702397 | 0.869779601 | 0.526226 |
| MultinomialNB_alpha_1e-05: | 0.825277261 | 0.769982503 | 0.807565364 | 0.869745336 | 0.525747 |
| MultinomialNB_alpha_0: | 0.8245921 | 0.769742695 | 0.807599622 | 0.869745336 | 0.525267 |

The following table shows the results of 5-fold cross validation for hyperparameter tuning for RandomForest with parameter n_estimators set to the values shown. It was competitive for all cutoffs, and made the baseline F1 score for all as well.

| **Classifier used in 5-fold cross-validation** | **Cutoff 1 mean** | **Cutoff 2 mean** | **Cutoff 3 mean** | **Cutoff 4 mean** | **Multiclass mean** |
|---|---|---|---|---|---|
| RandomForest_50: | 0.851245662 | 0.802632044 | 0.814691279 | 0.876117881 | 0.530954 |
| RandomForest_75: | 0.849429855 | 0.808387457 | 0.813766252 | 0.878070657 | 0.539588 |
| RandomForest_100: | 0.850697559 | 0.809004383 | 0.815650365 | 0.877762326 | 0.5388 |
| RandomForest_125: | 0.850594684 | 0.810682864 | 0.818185592 | 0.877865148 | 0.547399 |
| RandomForest_150: | 0.850868774 | 0.811676466 | 0.818288325 | 0.877351232 | 0.548975 |
| RandomForest_175: | 0.850560391 | 0.810340316 | 0.81900782 | 0.876974362 | 0.551373 |

The following table shows the results of 5-fold cross validation for hyperparameter tuning for Adaboost (adaptive boost) with parameter n_estimators set to the values shown. It was competitive for all cutoffs, and made the baseline F1 score for all as well.

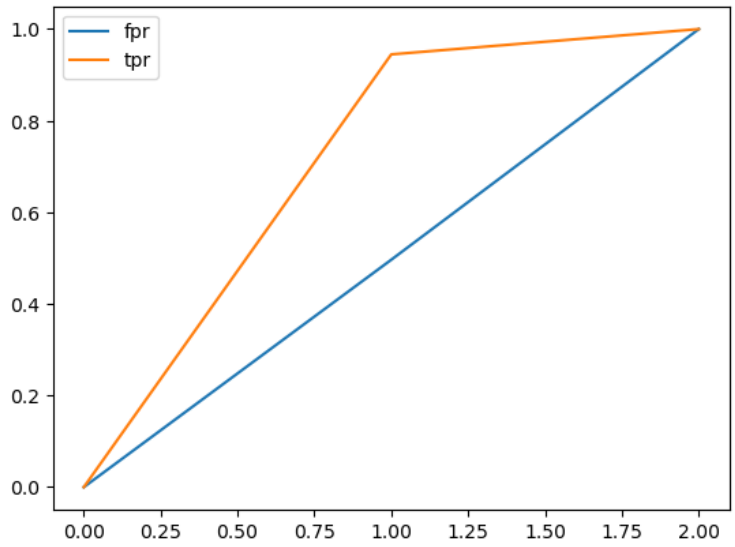
| **Classifier used in 5-fold cross-validation** | **Cutoff 1 mean** | **Cutoff 2 mean** | **Cutoff 3 mean** | **Cutoff 4 mean** | **Mul** |
|---|---|---|---|---|---|
| Adaboost_50: | 0.848196638 | 0.778890227 | 0.794340964 | 0.87858435 | |
| Adaboost_75: | 0.848744865 | 0.787695046 | 0.802803195 | 0.878173279 | |
| Adaboost_100: | 0.852958789 | 0.793656027 | 0.811642167 | 0.880400192 | |
| Adaboost_125: | 0.852582012 | 0.796876466 | 0.817192124 | 0.883106699 | |
| Adaboost_150: | 0.852958824 | 0.798246758 | 0.819007949 | 0.882284376 | |
| Adaboost_175: | 0.853404253 | 0.799069068 | 0.820926415 | 0.88063986 | |

The final table shows the results of 5-fold cross validation for hyperparameter tuning for Logistic Regression with parameter solver set to the values shown. The liblinear solver performed better for Cutoff 1 but worse with Cutoff 3 and Cutoff 4.

| **Classifier used in 5-fold cross-validation** | **Cutoff 1 mean** | **Cutoff 2 mean** | **Cutoff 3 mean** | **Cutoff 4 mean** |
|---|---|---|---|---|
| LogisticRegression: | 0.802151283 | 0.680946113 | 0.689298015 | 0.843709111 |
| LogisticRegression_liblinear_42: | 0.805131756 | 0.770155508 | 0.666279959 | 0.807290399 |

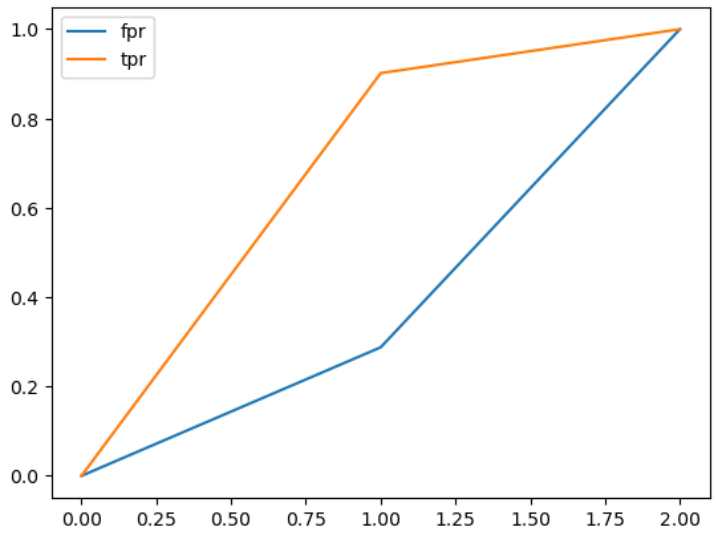
### *Results for Binary Classification with Cutoff 1*

The winning classifier was Adaboost with hyperparameter n_estimators set to 175. The ROC curve is below, with AUC score 0.71. This was the worst score of all the binary classification tasks. Other scores are available in the Notebook.



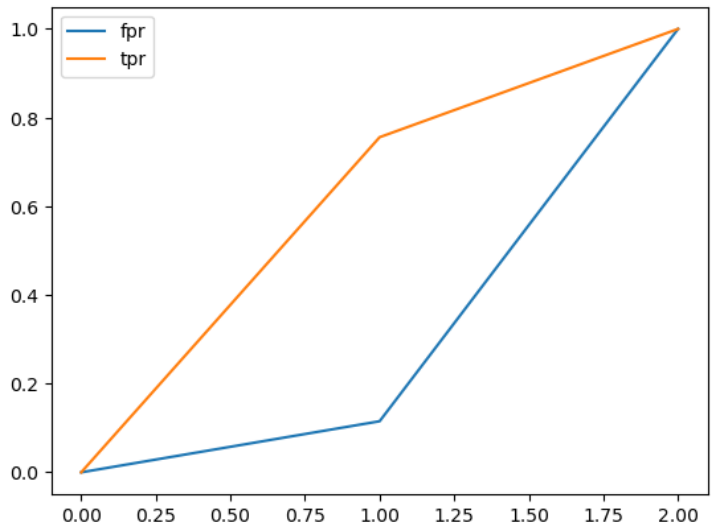
***Results for Binary Classification with Cutoff 2***

The winning classifier from cross-validatoin was RandomForest with hyperparameter n_estimators set to 150. The ROC curve is below with AUC score 0.81. Other scores are available in the Notebook.



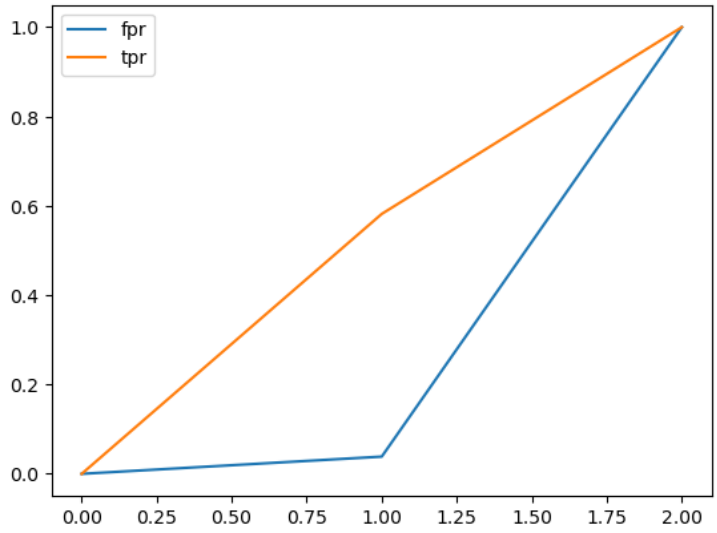
***Results for Binary Classification  with Cutoff 3***

The winning classifier from cross-validation for cutoff 3 was Adaboost with hyperparameter n_estimators set to 175. The AUC score was 0.82, similar to cutoff 2. Other scores are available in the Notebook.



***Results for Binary Classification with Cutoff 4***

The winning classifier from cross-validation for cutoff 4 was Adaboost with hyperparameter n_estimators set to 125. The roc score was 0.77. Other scores are available in the Notebook.




***Results for Multiclass Classification***

The ROC curve is not applicable to multiclass specification. The winning classifier from cross-validation was RandomForest with hyperparameter n_estimators set to 175.

***Results for Clustering***

After experimenting with many different values for max_features when vectoring the text in the review text and review summary, I found that value 6 provided the best silhouette score of 0.63, better than the baseline score of 0.59.

Citations

[1] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[2] Foreman, J. *Data Smart* Wiley, 2014. Print.

[3] Monir Yahya Ali Salmony; Arman Rasool Faridi, Supervised Sentiment Analysis on Amazon Product Reviews: A survey 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)

[4] Tyler Doll, Matthew Bussing, Kai Nichols, Sidney Johnson, A Machine Learning Approach to Automated Customer Satisfaction Surveys (medium.com)