Multiclass Classifier Building with Amazon Data to Classify Customer Reviews into Product Categories

Yunzhen Hu
yzhu14@stanford.edu

Te Hu
teh@stanford.edu

Haier Liu
haierliu@stanford.edu

Abstract

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 which could help promote commercial prediction and advertisement.[1]

Our project aims to build a multiclass classifier based on the customer reviews which are labeled by product categories in Amazon. Different feature selections and models are tested when building this classifier. As we will show below, the random forest classifier performs the best.

I. Introduction

To make predictions about potential customers' needs and thus feed him/her relevant advertisement is one of the prevailing applications of machine learning. For example, some person posts something online, say, a tweet expressing his/her interest in a pair of skateboard shoes. If a certain classifier could recognize this potential need and then display several brands of skateboard shoes to the Twitter user, there could be a good chance that this deal is made.

To build such a classifier, our project both trains and tests on customer reviews form Amazon. We select Amazon as our data source since it has a detailed and clear product hierarchy and also has plenty of customer reviews as samples.

II. Dataset

Our data is provided by our project advisors Prof. Ashutosh Saxena and Mr. Aditya Jami from SNAP.

Figure 1 Amazon product hierarchy

Figure 1 shows the structure and classification of goods in Amazon. There are 25 root categories and further subcategories belonging to each of them. We only train and test on the top level categories in our project.

We also have a large dataset related to 137,623 items in Amazon. Information contained includes description of each item, customer reviews, relations to above categories, etc.

[1] The net sales of Amazon is $74.452 billion in 2013 and $61,093 billion in 2012. "AMAZON COM INC 2013 Annual Report Form (10-K)". United States Securities and Exchange Commission. January 31, 2014.

III. Features & Preprocessing

We extract 1,276,443 customer reviews from the raw dataset. After labeling these reviews by their corresponding root parent categories, we construct a sample with 1,000 reviews for each of the 21 root categories. (The other 4 categories are not included since there are not enough reviews for them.)

Tokens are generated from these 21,000 reviews, before which they are all converted into lowercase and stemmed by Porter Stemming Algorithm. With the Bag-of-Words Model, a dictionary of all the 18,455 tokens is constructed and each review is transformed into a Boolean-valued $1 \times 18{,}455$ vector. (An element is labeled 1 if its corresponding token appears in the review and 0 otherwise.)

Two feature filters as shown below, Mutual Information (MI) and Term Frequency – Inverse Document Frequency (TF-IDF), are built to calculate feature scores. Different thresholds are tested in running the classifiers in order to decide the best feature sizes for different models.

$$MI_k = \sum_{x_k=0,1} \sum_{y=0}^{20} P(x_k, y) \log \frac{P(x_k, y)}{P(x_k)P(y)}.$$

$$TF - IDF_k = \max_{y=0,\ldots,20} P(x_k, y) \log \frac{1}{P(x_k)},$$

Then we split the 21,000 reviews into training and testing samples. For each root category, the training size is 700 and the testing size is 300.

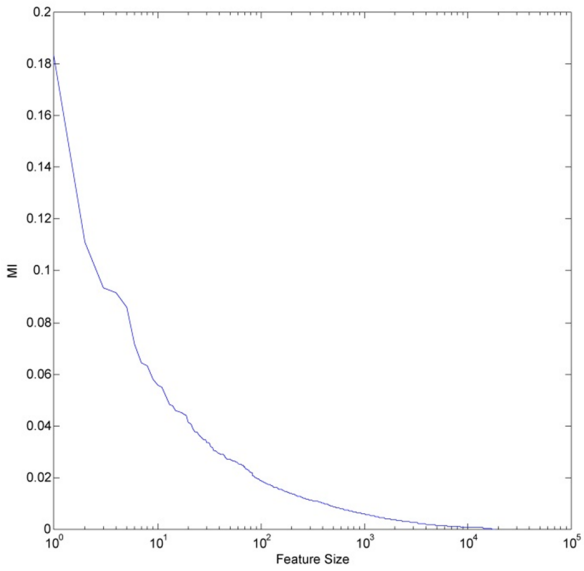

Figure 2 Distribution of MI scores of tokens

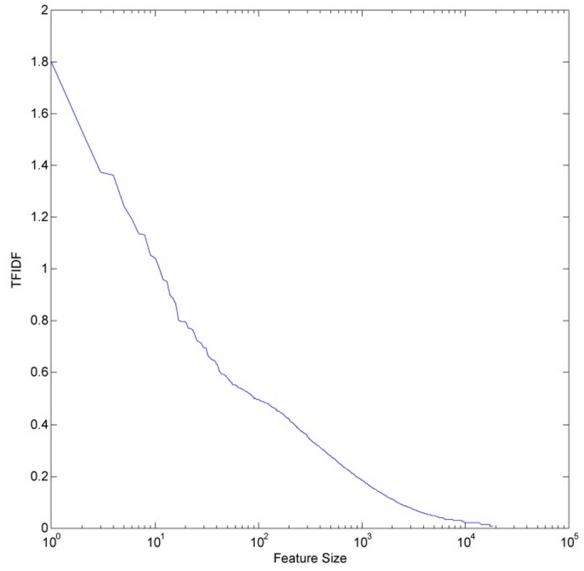

Figure 3 Distribution of TFIDF scores of tokens

IV. Models

We have built 5 multiclass classifiers: Random Forest Classifier (RF), Decision Trees Classifier (DT), Linear Discriminant Analysis Classifier (LDA), Linear Supporting Vector Classifier (SVC) and Nearest Centroid Classifier (NC).

Decision Trees Classifier builds a decision tree according to the information gain for each attributes. It chooses the attribute with the largest information gain at each node, and stops adding node when every attribute has already been included along this path through the tree, or the training examples associated with this leaf node all have the same target attribute value.

Random Forest Classifier repeatedly selects, by bagging, a random sample with replacement of the training set and fit 1000 trees to the samples. With each tree-fitting with an individual decision trees classifier, the random forest classifier takes the majority vote in the case of decision trees.

Linear Discriminant Analysis Classifier sets a common covariance matrix for conditional distributions given different labels.

$$f(x) = arg \max_{y=1,\dots,k} P(X = x|Y = y) \cdot P(Y = y);$$

$$P(X|Y = k) \sim N(\mu_k, \Sigma).$$

Linear Support Vector Classifier applies the SVM algorithm.

Nearest Centroid simply decide the labels by computing and find the nearest centroid.

$$\vec{\mu}_k = \frac{\sum_i \vec{x}^{(i)} 1(y^{(i)} = k)}{\sum_i 1(y^{(i)} = k)};$$

$$\hat{y} = arg \min_{k \in Y} \|\vec{\mu}_k - \vec{x}\|.$$

V. Results

In Table 1 and 2 we show the results of different models with features filtered by MI and TFIDF, respectively. Figure 4 and 5 are also shown for readers' easier understanding of the trend between the test accuracy and the score thresholds.

Table 1 Model performances with features filtered by MI

MI	RF	DT	LDA	SVC	NC
0.01	0.634	0.539	0.526	0.582	0.321
0.009	0.647	0.537	0.543	0.591	0.313
0.008	0.650	0.535	0.554	0.601	0.306
0.007	0.659	0.546	0.572	0.623	0.304
0.006	0.667	0.543	0.587	0.628	0.303
0.005	0.676	0.558	0.614	0.640	0.304
0.004	0.683	0.552	0.639	0.655	0.309
0.003	0.693	0.555	0.661	0.658	0.311
0.002	0.695	0.558	0.681	0.663	0.316
0.001	0.699	0.560	0.625	0.683	0.317

Table 2 Model performances with features filtered by TFIDF

TFIDF	RF	DT	LDA	SVC	NC
0.2	0.654	0.542	0.573	0.608	0.297
0.18	0.662	0.542	0.594	0.620	0.300
0.16	0.668	0.540	0.605	0.627	0.303
0.14	0.678	0.542	0.623	0.634	0.306
0.12	0.683	0.547	0.639	0.635	0.308
0.1	0.687	0.547	0.660	0.647	0.312
0.08	0.690	0.556	0.667	0.655	0.314
0.06	0.691	0.557	0.681	0.663	0.315
0.04	0.697	0.561	0.663	0.679	0.317
0.02	0.703	0.571	--	0.685	0.318

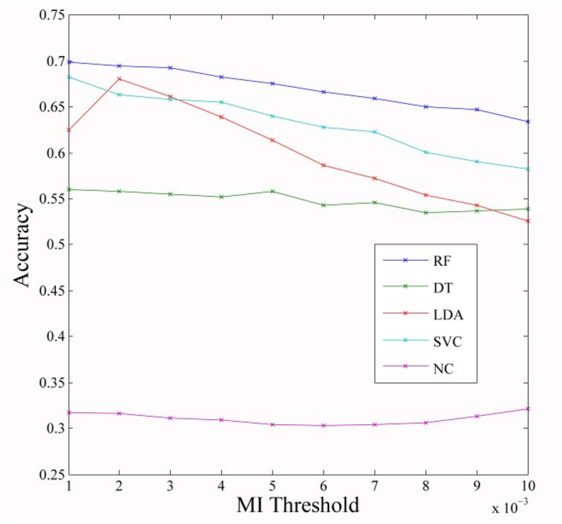

Figure 4 Model performances with features filtered by MI

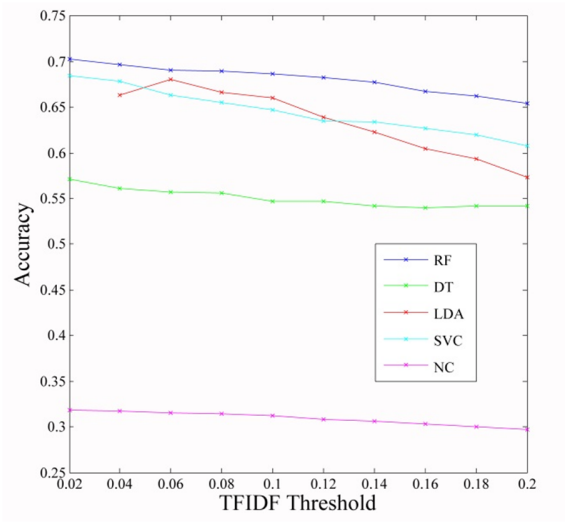

Figure 5 Model performances with features filtered by TFIDF

VI. Discussion

Generally, Random Forest Classifier performs the best. For all the feature sets, Random Forest Classifier gives the largest accuracy. The optimal value is 0.703 with a threshold of 0.02 (filtered by TFIDF).

When more features are included in the model, both Random Forest Classifier and Linear Supporting Vector Classifier perform better, which shows that these two models have a good capability of preventing over-fitting.

No evidence shows that the difference of feature filters would significantly influence the performance. With the number of features constant, say, around 3,780, the filters generate similar results for every model. With the threshold on feature scores lowering, Linear Discriminant Analysis Classifier's accuracy firstly increases and then decreases, which demonstrates that the filters are both valid. But since TFIDF values distribute more evenly than MI values of the features, the application of TFIDF may be more convenient.

VII. Conclusions

For this project, Random Forest Classifier performs the best (with 1000 estimators). The performances of Random Forest Classifier and Linear Supporting Vector Classifier will improve as the number of features increases.

Since Random Forest Classifier has the capability of preventing over-fitting, a threshold of 0.02 on TFIDF works fine. We prefer TFIDF to MI because TFIDF has a more evenly distribution.

VIII. Future

To improve the accuracy of the multiclass classifier, we are going to try bi-gram tokens. Moreover, a hierarchical classifier can be built on the dependences between labels at different levels. Also, cross-domain learning can be implemented with data from Twitter and E-Bay.

Acknowledgement

We would like to thank Professor Ashutosh Saxena and Mr. Aditya Jami, who have provided us with so many suggestions and data. We have learnt a lot from them in our regular meeting every Tuesday.

References

[1] Teh, Y. W., Jordan, M. I., Beal, M. J., & Blei, D. M. (2006). "Hierarchical dirichlet processes". *J. Amer. Stat. Assoc.*, *101*(476).

[2] Jiang, Y., & Saxena, A. (2013, August). "Discovering different types of topics: Factored topic models." *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence* (pp. 1429-1436). AAAI Press.

[3] Daumé III, H. (2009). "Frustratingly easy domain adaptation". *arXiv preprint arXiv:0*