Predicting Informative Amazon Reviews by Customers

Jayalakshmi Sureshkumar, Eihal Alowaisheq, Tan-Hsun Weng

1. Introduction
In this project we are trying to predict whether an online product review is informative or not. Uninformative reviews might indicate that this review may be a spam or an unhelpful review. This could be served as a prefiltering tool for webmaster so they can keep an eye on a mass of uninformative reviews.

2. Background
Semi-supervised learning falls in the middle of supervised learning and unsupervised learning. In supervised learning, we have labelled data, from which we learn the model, while in unsupervised learning, we have unlabelled data, which is partition based on similarity between feature vectors. Semi-supervised learning is to use partially labelled data, to improve the prediction of the unlabelled data. Consider an unsupervised method where we have some labelled data, we can weight the labelled data higher around the training point we don't know the clusters of, and form a better model.

Applying semi-supervised learning to text classification is preferred because acquisition of labeled training data is not as easy as that of getting unlabeled data. We need human intervention to label the given text document which is not only time consuming but also higher error rate. With the unlabelled dataset combined with labeled dataset, we can get a better performance classifier by training both together, which make up for the shortcomings of small dataset. The semi-supervised classification methods includes Self-Training, Transductive Learning, Generative Model, Disagreement-Based Methods, Generative Methods, Discriminative Methods, and Graph-Based Methods. Other methods are Semi-Supervised Regression, Semi-Supervised Clustering and Semi-Supervised Dimensionality Reduction.

Semi-supervised Clustering method is common for similarity measure and search for appropriate clusters for prediction. In this project, we chose semi-supervised clustering method with 100 labeled dataset for training and 1000 unlabeled dataset for clustering in order to get the results of predicting if the reviews are informative or not. We further increased the size of the unlabelled training set, and performed clustering.

3. Tools
1. Python with Gensim (python implementation for Word2vec), and Sklearn packages
2. Weka

4. Dataset
The dataset used for this project is the amazon review data for electronic devices. There are totally 143.7 million reviews in the data file.

The problem at hand is to predict the usefulness of a review, and we have data that is not labelled for the given problem.

The reviews were extracted from the data, with an assigned unique identifier for each review. Initial samples of data was taken from the entire dataset, by first extracting 1100 reviews. Since, we are taking a small sample from a huge data, we can not take the first ‘n’ reviews. Thus, to keep a mix of reviews, reviews of various length were taken. Then, we took 100 data points from the 1100 data points and manually labelled them. Thus, we have 100 labelled data and 1000 unlabelled data. Also, we have 50,000 unlabelled data.
So, our dataset could be summarize our dataset as follows:

1. Dataset A: 100 manually labeled reviews, contains around 1K words
2. Dataset B: 1000 unlabeled reviews, consists of around 1.5M words
3. Dataset C: around 50k unlabeled reviews contains about 5M words

Dataset A is primarily used as testing dataset, however in we used it as training dataset in some experiments.

5 Feature Selection and Feature Engineering

5.1 Bag-of-words (BOW):

Using BOW we created word vectors of 1000 features. Stop words have been removed from the features set, so the feature set will not contain very common words such as: “the”, “that”, ..etc. Then we generated frequency matrix for dataset A and B . Since informative reviews usually tend to be longer than uninformative review, we included the length of the review to the features. Thus, the total number of features is 1001 features.

5.2 Word2Vec (W2V)

We used Word2Vec with A, B, C datasets to generate average matrix. We ignored stop words as well and we used skip-gram architecture.
We tuned the parameter as follows:

1. # number of features: we tried different # of features 300 - 500 features. Increasing the number of features further do not improve the performance. That could because our dataset is relatively small
2. Minimum word count: we can specify the minimal number of occurrences of a word that should appear in the document to be considered in the vector. Based on our experiments Setting the minimum word count to values between 70-100 gave improved performance. Setting this parameter to small values leads to considering less important words in our vector, while setting it to very large number the algorithm will only consider words with very high frequencies.
3. Context/window size: the parameter is used to specify the number of words the algorithm uses to lean the context. Based on our experiment on our dataset 10-20 were reasonable values.

6. Clustering

6.1 Algorithm and Results

We performed two clustering algorithms, K-means and EM using different feature vectors that we generated in section 4, ran them on and datasets A, B, C. Table 1 shows the details and evaluation of our experiments:

Features & Dataset	**K-means**	**EM**
BOW A 1001 features	Accuracy: 63% Precision for informative: 61.45% Recall for informative: 100% Precision for uninformative: 100% Recall for uninformative: 9.75% AUC: 0.55 **(1)**	-
BOW Randomized and Normalized A 1001 features	Accuracy: 65% Precision for informative: 96.15% Recall for informative: 42.37% Precision for uninformative: 53.33% Recall for uninformative: 97.56% AUC: 0.70	-
BOW B Training + A Testing 1001 features	Accuracy: 85% Precision for informative: 97.82% Recall for informative: 76.27% Precision for uninformative: 74.07% Recall for uninformative: 97.56% AUC: 0.87	Accuracy: 76% Precision for informative: 72.15% Recall for informative: 96.61% Precision for uninformative: 90.47% Recall for uninformative: 46.34% AUC: 0.71
W2V A 300 features 40-100 min word count	Accuracy: 62% Precision for informative: 71.42% Recall for informative: 59.32% Precision for uninformative: 52.94% Recall for uninformative: 65.85% AUC: 0.64 **(2)**	-
W2V B Training + A Testing 300 features 40-100 min word count	Accuracy: 53% Precision for informative: 62.5% Recall for informative: 50.84% Precision for uninformative: 44.23% Recall for uninformative: 56.09% AUC: 0.53	Accuracy: 49% Precision for informative: 57.40% Recall for informative: 52.54% Precision for uninformative: 39.13% Recall for uninformative: 43.90% AUC 0.48 **(3)**
W2V Randomized B Training + A Testing 300 features 40-100 min word count	-	Accuracy 69% Precision for informative: 91.11% Recall for informative: 52.54% Precision for uninformative: 57.57% Recall for uninformative: 92.68% AUC: 0.73 **(4)**
W2V Randomized C Training + A Testing 500 features 100 min word count	Accuracy: 61% Precision for informative: 63.15% Recall for informative: 81.35% Precision for uninformative: 54.16% Recall for uninformative: 31.70% AUC: 0.57	Accuracy: 77% Precision for informative: 71.95% Recall for informative: 100% Precision for uninformative: 100% Recall for uninformative: 43.90% AUC: 0.72 **(4)**

Table 1: Performance Evaluation of applying K-means and EM on different datasets

As we can see in Table 1 the best performance was achieved when using the EM with randomized C (relatively large dataset).

6.2 Analysis and Discussion

BOW:

1. In BOW the normalization and randomization improve the performance even with increasing the dataset size. This is expected since normalization scales the words frequencies and randomization helps providing good initial seeds for K-means
2. In BOW adding the length of the review was a good measure to predict whether a review is informative or not
3. Also, increasing the size of the data yields better result, and prevents underfitting.
4. With BOW, the best performance was observed with 1001 features with k means. Though k means gave better overall accuracy on the test set, EM seem to give a better model in terms of predicting informative reviews. The high recall for informative reviews and high precision for uninformative reviews for EM (BOW B Training + A Testing with EM in Table 1) indicates that there is a good amount of correctly classified informative reviews.

W2V:

1. Randomization helps improving the performance of EM (as we can see in (3) and (4) in Table 1). This proves the concept that the initial seeds affects the quality of the clustering algorithm
2. Increasing the number of datapoints definitely improves the performance. Although Google recommends using tens of billions words, the largest dataset the we used was C which includes only around 5M words and we were able to reach 77% accuracy in EM and 61% in K-means.
3. In general W2V performs better than BOW. We can see in (1) and (2) in Table 1 that even when using small dataset and using less features in W2V (only 300 compared to 1001 in BOW), W2V outperforms BOW. That is because W2V considers the semantic relationships among the words when selecting features.
4. The purple line in Figure 2, and the evaluation information (4) in Table 1 shows us that the best outcome was from EM with 50,000 reviews. The results in the table also show us a similar pattern as in the case of BOW, that the informative reviews were well identified by EM in this case.
5. In most of our experiment EM outperforms K-means. This could be because EM provide soft assignment of data point to clusters with a probabilistic approach, while K-means makes hard assignments.

Figure 1: Area under ROC with Bag of Words Features

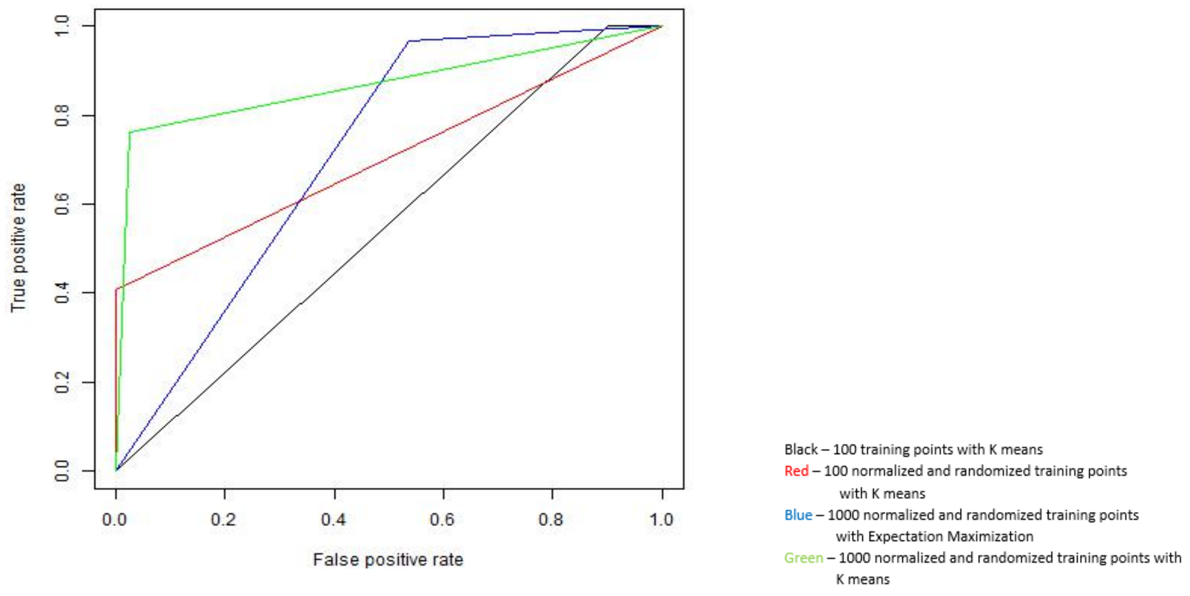
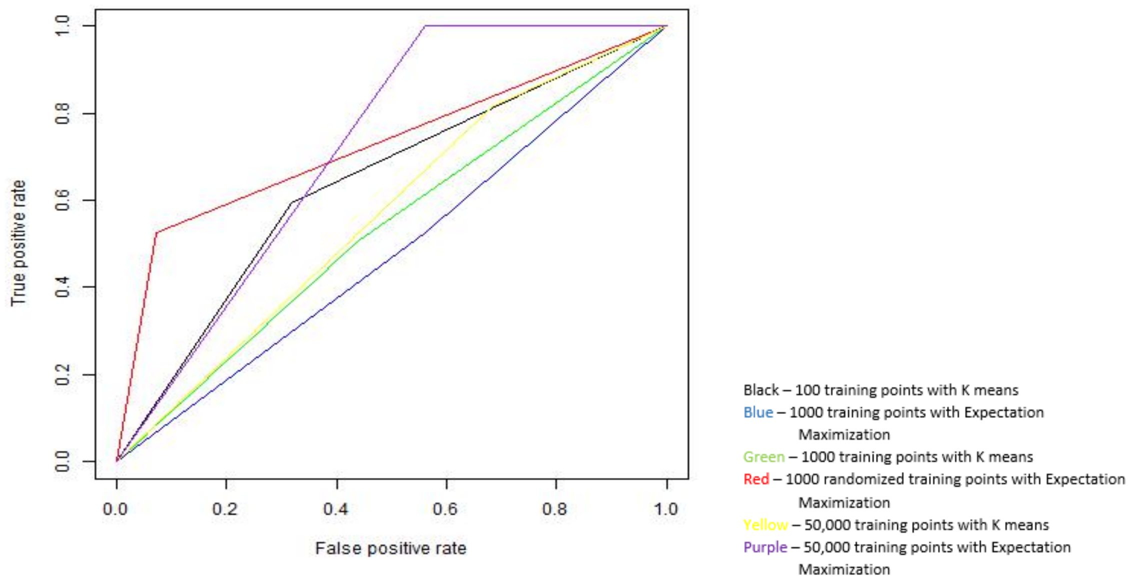


Figure 2: Area under ROC with Word2Vec Features



7. Classification

7.1 Algorithm and Results

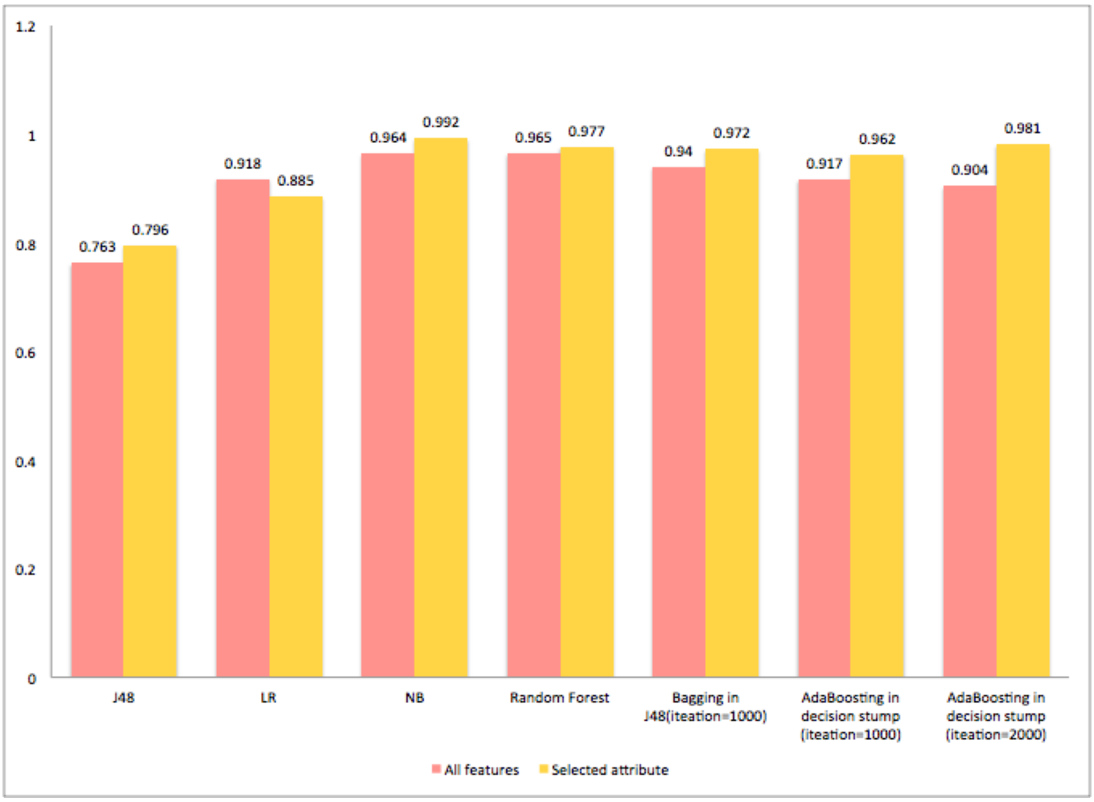
We used 100 Labeled reviews with 10-fold cross-validation for training. In order to see the accuracy of the training dataset, we used six different learning algorithms with different parameter for training 1001 features and 37 selected features respectively. The learning algorithms included Naive Bayes, Logistic Regression, Decision tree J48, Random forest, Bagging in J48 with 1000 iteration, AdaBoosting in decision stump with 1000 iteration, and

AdaBoosing in decision stump with 2000 iteration. We compared the AUC ROC and the results showed in Table 2.

Table 2: classification results

Area under ROC (Cross-validation)	J48	Logistic Regression	Naive Bayes	Random Forest	Bagging in J48(iteration=1000)	AdaBoosting in decision stump (iteration=1000)	AdaBoosting in decision stump (iteration=2000)
All features	0.763	0.918	0.964	0.965	0.94	0.917	0.904
Selected attribute	0.796	0.885	0.992	0.977	0.972	0.962	0.981

Figure 3: Classification Results



In training dataset with all features (1000 features and feature of length), it showed that Random forest (0.965), Naive Bayes(0.964), and Bagging(0.94) have higher accuracy rate than the others.

In training dataset with 37 selected features by Weka, the results showed that it had higher AUC than results of training 1001 features at each classifiers. Naive Bayes and Adaboosting in decision stump with 2000 iteration had higher accuracy, 0.992 and 0.981 respectively.

We also tested 1000 features without feature of length in order to see the difference, but the results showed that there were no difference in AUC at each classifier.

7.2 Analysis and Discussion

1. Naive Bayes shows better performance, 0.964 AUC and 0.992 AUC, than others because of the dataset is too small. In this condition, we did not consider Naive Bayes is our better classifier.
2. In training dataset with all features (1000 features and feature of length), random forest or Bagging in J48 with 1000 iteration is our better choice, however, since we our labeled dataset is small, along with large features in training, it would have higher variance.
3. In training dataset with 37 selected features by Weka, the reason it showed that there was higher AUC than training dataset with all features is that reducing features also increases bias, in this case, AdaBoosting is our better choice.
4. Feature of length of the review didn't affect the AUC in classification.

8. Conclusion

We found that the general performance of classification is better than clustering, and this is inherent because of the property of classification to learn from the labels. Classification is analogous to figuring out the question when an answer is given, which makes the quality of the question better.

Classification performed better without the length of the review, while with clustering length of the review seem to help improve the cluster quality. AdaBoost with 37 features gave us the best results in classification. Initially, there was less data with high dimensionality, and boosting with feature reduction reduced bias in this scenario and improved the results.

Given the opportunity to use a huge amount of data, but finding that the data is unlabelled, it is better to use semi-supervised or unsupervised learning, as more data is always good. It provides us with much more information than we would have with a small sample of the data.

The features are crucial when we perform clustering. Since we don't learn a model based on the patterns between class and the features, we need particular features that would give us good clusters with respect to the problem that we are trying to solve. With good features, that are scaled and with a data that can provide us with good initial seeds, results improve considerably.

Expectation Maximization generally gave us better results than K means. This is because it performs soft assignment using a probabilistic approach. This counters possible overfitting in training data and provides better results with the test data.

9. References

1. Bag of Words Meets Bags of Popcorn, , https://www.kaggle.com/c/word2vec-nlp-tutorial/details/part-2-word-vectors, June 2015
2. Semi-supervised learning. 2006

3. Alldrin, N., Smith, A., & Turnbull, D. Clustering With EM and K-Means.
4. Bair, E. (2013). Semi-supervised clustering methods. Wiley Interdisciplinary Reviews. Computational Statistics, 5(5), 349–361.
5. Image-based recommendations on styles and substitutes J. McAuley, C. Targett, J. Shi, A. van den Hengel SIGIR, 2015 pdf
6. Inferring networks of substitutable and complementary products J. McAuley, R. Pandey, J. Leskovec Knowledge Discovery and Data Mining, 2015 pdf