# Predicting Natural Language Sentiment using Machine Learning Classifiers: A Survey

**Mayank Mahajan, NetID: mmahajan**
Princeton University
mmahajan@princeton.edu

## Abstract

With such large amounts of text reviews being written everyday, performing sentiment analysis to quantitatively classify these quotes has become a increasingly difficult, yet still important task. We explored different classes of classifiers in their ability to accurately predict the class for a sample of reviews. After performing decision tree feature selection and 4-fold cross validation, the Neural Network and RBF SVM were found to achieve the best accuracy (74%) and the best $F_1$ scores (68%). The Multinomial NB was found to be the fastest classifier (0.02 sec) with one of the most accurate (73.4%) and specific/sensitive ($F_1 = 68\%$). The classifiers also found that words with positive and negative semantic meaning are the most discriminative in determining the overall class of the review. These results show how specific classifiers can be used to create a fast, accurate, and precise system for sentiment analysis with a highly reduced set of features.

## 1 Introduction

The rise of online reviews across the entertainment, shopping, and dining industries have led to a proliferation of textual data rating different products. Beyond the numerical scores that often accompany these reviews, the text itself contains unique insights that a basic number cannot describe. It is then of certain value to extract such insights from this text data, which can be used for a myriad of applications, from aiming to understand a specific user's sentiment toward an item, to holistically predicting what overall user sentiment toward a product might be, especially in cases where no quantitative score currently exists. For example, this type of natural language analysis can drive key business insights for providers of goods or services to improve their operations to reach a satisfactory level, or give feedback for a new product that has just launched.

Many methods currently exist to analyze this type of data, the most famous of which have been arduously studied by technology companies like Google[7] or Facebook[3]. Machine learning, in particular, is an area of computer science that has emerged as a dominant force in the task of identifying features of text that can reliably signal sentiment information about a particular sample. Yet, there does not exist a definite machine learning approach for sentiment analysis, particularly for review data. Considering the degree of variation that can occur within the reviews for a specific item, much less a set of items, the optimal classifier is not always obvious and is often dependent on the data itself. Due to the numerous machine learning methods that exist to analyze and classify text data, multiple methods should be compared to 1) find an accurate classification system for review data in general, and 2) to identify features of a specific dataset that prove to be useful across multiple techniques.

## 2 Related Work

### 2.1 Feature Selection

Whenever a large amount of features is involved, as is often the case with text classification, it is important to extract the most discriminative features for classification. In the reduced case where there are only two classes, positive and negative, features that show the most promise in differentiating a positive sample from a negative sample should be retained, and features that ostensibly have no bearing should be discarded. Ratanamahatana et al. shows how a decision tree can be a useful method of feature selection [4]. This method was also utilized for this experiment.

### 2.2 Classification

Predicting sentiment on text data is a known problem, and there have been several previous attempts to apply machine learning methods in order to learn an accurate classifier. One of the most common approaches has been to use a Nave Bayes Classifier, which applies Bayes' theorem to compute conditional probabilities for different sentiment classes, most likely positive or negative [8], [2]. Specifically for document classification, the Multinomial Naive Bayes classifier has been particularly useful in computing a likelihood from a bag-of-words vector of frequencies. While intuitive through their direct connection from vectors of data to likelihoods for each class, a strong confounding factor in Naive Bayes classification is conditional independence, i.e. the assumption that the value of any given feature is independent, given the class. In the case of review sentiment analysis, it posits that the presence of any given word relies solely on the class and is independent from the presence of any other word.

A more modern approach to classification has come from the field of deep learning, where networks that are modeled after the neural connections in human brains are used to receive several inputs and through various layers of nodes, output a result close to 1 for a positive result and 0 for a negative result.[9] These networks are trained to tune the hyperparameters to best predict sentiment, and each of the test vectors, where each input node corresponds to the presence or absence of a particular word in the review, will be fed through the network to generate a response.

Several other methods have been applied to text classification, namely SVMs, Random Forests[5], Adaboost Classifiers[1], and Decision Trees[4]. Previous results have shown that Naive Bayes and SVM classifiers have outperformed other classifiers, yet for the purpose of of thoroughness we present a comprehensive comparison of a variety of machine learning classifiers.

## 3 Methods

### 3.1 Feature Selection: A Decision Tree In Detail

One of the largest perils when working with a large lexicon is the curse of dimensionality and computational load that accompanies classification in high dimensions. Especially if the aim of a classifier is to assign a class to unlabeled data, it is important to avoid overfitting a classifier to the training data, which can be evidenced by a low generalization error. Particularly in the case of text classification with a bag-of-words representation, there is a significant risk that words appearing often in the training set might be overemphasized, which can lead to a loss of accuracy when new samples contain a large proportion of previously unseen words. Therefore, we aim to reduce the number of features/words in this problem.

To reduce the number of features, a decision tree was used to determine the features with the most discriminative power. The decision tree is an intuitive way to reduce the number of features: it involves an algorithm where the data is recursively split by a threshold value of the feature with the most information gain, which is a decrease in entropy. Entropy is a function of a random variable $Y$ with probability distribution $p = (p_1, p_2 \dots p_n)$ that measures the amount of information in a single message. The Shannon entropy, denoted by $H(p)$, calculates entropy using the log probabilities of the distribution. [6]

$$H(Y) = \Sigma_{i=1}^{n} - p_Y(y_i) \log p_Y(y_i) \tag{1}$$

Although the Shannon entropy measures the current variation in the data, we can also compute the conditional entropy, which determines how uncertainty there exists given the knowledge of a certain

of a feature setting. The conditional entropy of a discrete random variable $Y$, given another discrete random variable $X$, is computed by

$$H(Y|X) = -\Sigma_{i=1}^{n} p_X(x_i) \Sigma_{j=1}^{n} p_{Y|X}(y_j|x_i) \log p_{Y|X}(y_j|x_i) \tag{2}$$

Computing the difference between the entropy and conditional entropy yields a value known as the information gain, which measures how much information we learn about a variable $Y$ by learning the value of the random variable $X$.

$$IG(X) = H(Y) - H(Y|X) \tag{3}$$

The decision tree takes advantage of these values to determine which feature to split the data by at each stage of the algorithm. At each node, the algorithm computes an optimal feature and threshold on which to separate the data to maximize the information gain, and this process continues until the class of the data is completely determined by the splits in the tree.

Because this process can lead to an extremely large tree, the decision tree can be pruned at a certain depth so that only the most important features are retained. For this experiment, different maximum depths of the tree were tested until the total importance proportions for each feature surpassed 99%. The decision tree reduced the number of features from 3501 to 80, with 2% of features and a total feature importance of 99.5%.

One potential weakness of using this method is multicollinearity, where different features are highly correlated and lead to very similar information gains. In this case, the splitting algorithm's decision to use one feature over another becomes much more arbitrary and can lead to unwanted pruning. However, it remains to be seen whether this caveat will have a large impact on the accuracy of the classifier.

### 3.2 Classifiers

The classifiers for this study were chosen largely from previous related work on sentiment analysis. The classifiers chosen were:

1. K-Nearest Neighbors (K = 3)
2. Linear SVM (C = 0.025)
3. RBF SVM ($\gamma$ = 0.1, C = 10)
4. Decision Tree Classifier (depth=5)
5. Random Forest Classifier (depth=5)
6. Neural Net/Multi-Layer Perceptron Classifier ($\alpha$ = 0.1)
7. AdaBoost Classifier
8. Multinomial Naive Bayes Classifier ($\alpha$ = 0.0001)
9. Quadratic Discriminant Analysis

For each each of these classifiers, a Grid Search was performed with cross-validation to tune the hyper parameters to optimize the classifier before any training or testing took place.

### 3.3 Evaluation

To prevent overfitting to a particular training and test set, 4-fold cross validation was used to test each classifier. 4 folds were chosen to ensure that each test set would be exactly 600 samples out of the given data, which totaled 2400 samples. The precision, recall and $F_1$ score were measured to determine the degree of specificity and sensitivity of the classifiers. Given the number of false positives (FP), false negatives (FN), true positives (TP), true negatives (TN), and the false discovery rate (FDR), we can define the precision as the proportion of reported positives that are true positives, and the recall as the proportion of the positive samples that are reported as positive. Expressed quantitatively,

$$P = \frac{TP}{TP + FP} \qquad\qquad R = \frac{TP}{TP + FN}$$

The $F_1$ score is the harmonic mean of the precision and the recall, which aims to take an average of the two rates.

$$F_1 = 2\frac{P * R}{P + R}$$

All three measures were measured and analyzed in this experiment.

## 4 Results

| Name | Accuracy | ROC Area | Avg. Precision | Avg. Recall | $F_1$ | Time (s) |
|---|---|---|---|---|---|---|
| Neural Net | 0.7429 | 0.8527 | 0.87 | 0.5692 | 0.6882 | 6.4442 |
| RBF SVM | 0.7367 | 0.8151 | 0.8542 | 0.5692 | 0.6832 | 5.9548 |
| AdaBoost | 0.7354 | 0.8037 | 0.8687 | 0.5542 | 0.6767 | 0.8619 |
| Multinomial Naive Bayes | 0.7342 | 0.8146 | 0.8531 | 0.5642 | 0.6792 | 0.0218 |
| Nearest Neighbors | 0.6829 | 0.7911 | 0.8357 | 0.4558 | 0.5899 | 0.5103 |
| QDA | 0.6754 | 0.7856 | 0.7461 | 0.625 | 0.6802 | 0.059 |
| Linear SVM | 0.66 | 0.7921 | 0.8607 | 0.3758 | 0.5232 | 6.4895 |
| Decision Tree | 0.6396 | 0.6698 | 0.8677 | 0.3258 | 0.4737 | 0.0354 |
| Random Forest | 0.6225 | 0.7039 | 0.631 | 0.6625 | 0.6464 | 0.2465 |

Table 1. Cross-Validation Results for Different Machine Learning Classifiers

The accuracy and ROC area under the curve (AUC) scores for each tested classifier are listed above. To compare the discriminative power of each classification method, ROC curve graphs were plotted by varying the threshold parameter.

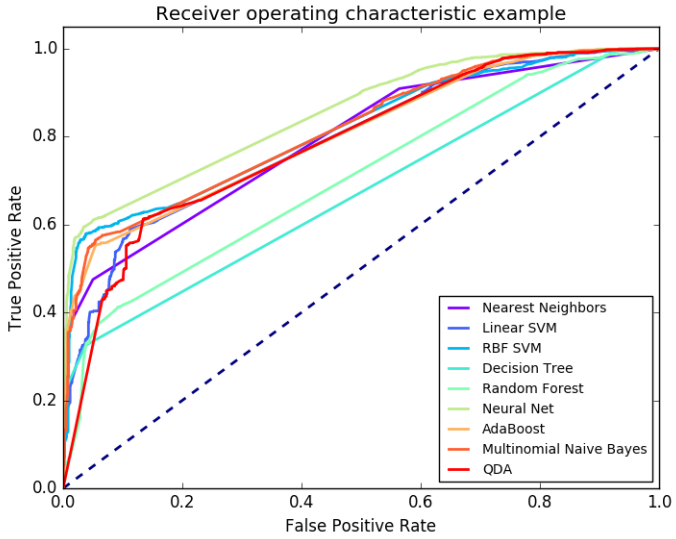


Figure 1. ROC Curves for Different Machine Learning Classifers

## 5 Discussion and Conclusion

Looking at the raw accuracy of all of the classifiers, it seems that the historically promising methods like the Linear Support Vector Machine and the Decision Tree Classifier actually underperform relative to the other classifiers, which is a departure from the literature. Instead, certain methods like the Neural Net and the RBF SVM do a better job of predicting sentiment given a vector of words. The accuracy attained for both of these methods is approximately 74%, which is quite impressive considering the classifier uses only 2% of the original features. The AdaBoost and Multinomial NB Classifiers fall close behind in terms of accuracy, but other methods are significantly behind.

Not only are the Neural Net and the RBF SVM the most accurate of the tested classifiers, but they also have one of the highest areas under the ROC curve, signalling its high probability of assigning

a higher score to a random positive sample than a random negative sample. There actually seems to be a decently strong positive correlation between the accuracy of the classifier and the area under the ROC curve; as the accuracy increases, the ROC area also tends to increase.

Investigating the precision and recall of the classifiers allows for a closer analysis of determining the specificity and sensitivity of the classifiers. Almost all of the classifiers achieved a similar precision rate (85%), with the exception of the Random Forest and the QDA Classifier. The majority of the classifiers were quite performant in returning true positive samples, meaning that positive samples could be trusted to a high degree. The recall, on the other hand, was significantly lower for all methods. Surprisingly, the same classifiers that had poor precision attained a higher recall than any of the other methods. The Random Forest and QDA both achieved recall rates of over 62%, while the Neural Net and the RBF SVM only achieved a rate of 56.92%. On the lower end of the spectrum, the Linear SVM and the Decision Tree performed extremely poorly, with recall rates of ¡ 40%, meaning they did not capture even half of the positive samples.

To reconcile the precision and recall rates, we computed the $F_1$ scores for each classifier. At the top end, the Neural Net, RBF SVM, and Multinomial NB all attained an $F_1$ score of around 68%, so it seems that by taking both specificity and sensitivity into account, all three of these classifiers perform similarly. It is worth noting that the $F_1$ score does not take into account the accuracy of the classifier, so to maximize the overall performance, the Neural Net was still the best method.

Lastly, it is of interest to look at the time to train each of the classifiers, as an indicator of the classifier's ability to scale efficiently with the amount of training data or number of features. In this respect, the Neural Net and the RBF SVM actually took much longer (around 6 sec) than the Multinomial NB classifier (0.02 sec). Because of the Multinomial Naive Bayes's respectable performance and its extremely fast training, it is a very good option for larger datasets or any scenario where time is an important factor.

The top 10 features for classification, measured with the ANOVA F-value between label/feature for classification tasks, were:

['bad', 'best', 'delici', 'excel', 'good', 'great', 'love', 'minut', 'nice', 'poor'].

From a semantic perspective, it is clear to see that these words would be good indicators of a positive or negative review. This encourages the notion that a review can easily be characterized just by looking for individual positive or negative words instead of needing the full text of the review, and that the classification of a review is intuitively understandable by a human.

In this experiment, we compared different classes of classifiers in their ability to predict sentiment labels for a dataset of different reviews. We performed feature selection and found that, under cross-validation, the neural network and the RBF SVM attained the highest level of accuracy at (74%), and the highest $F_1$ scores as well at (68%). These two classifiers were also some of the slowest classifiers (around 6 sec), and the Multinomial Naive Bayes classifier achieved almost the same level of results with a significantly faster runtime (0.02 sec). The experiment also confirmed that semantically positive and negative words were the most useful in predicting the sentiment of a review.

**Acknowledgments**

The author would like to thank Barbara Engelhardt and Xiaoyan Li for advising him during the writing of this paper.

## References

[1] Jianfang Cao, Junjie Chen, and Haifang Li. An adaboost-backpropagation neural network for automated image sentiment classification. In *TheScientificWorldJournal*, 2014.

[2] Neha Gupta and Shabnam Parveen. Efficient sentiment analysis using optimal feature and bayesian classifier. In *International Journal of Computer Applications*, pages 10–14, 2016.

[3] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[4] Olga Kolchyna, Thársis T. P. Souza, Philip C. Treleaven, and Tomaso Aste. Twitter sentiment analysis. *CoRR*, abs/1507.00955, 2015.

[5] Hitesh Parmar, Sanjay Bhanderi, and Glory Shah. Sentiment mining of movie reviews using random forest with tuned hyperparameters.

[6] Kohtaro Tadaki. The tsallis entropy and the shannon entropy of a universal probability. *CoRR*, abs/0805.0154, 2008.

[7] Chenhao Tan, Ed. H Chi, David Huffaker, Gueorgi Kossinets, and Alexander J Smola. Instant foodie: Predicting expert ratings from grassroots. In *Proceedings of the 22nd ACM international conference on Information  Knowledge Management*, pages 1127–1136, 2013.

[8] Yun Xu, Xinhui Wu, and Qinxia Wang. Sentiment analysis of yelp's ratings based on text reviews. Stanford University.

[9] Xiang Zhang and Yann LeCun. Text understanding from scratch. *CoRR*, abs/1502.01710, 2015.