# Machine Learning based classification for Sentimental analysis of IMDb reviews

Chun-Liang Wu
Stanford University
wu0818@stanford.edu

Song-Ling Shin
Stanford University
shin0711@stanford.edu

## 1. Introduction

In this big-data era, machine learning is a trending research field. Machine learning enables data analytics to study massive data in an effective way. This technique is very helpful to classify and predict the content of the language [1], which is also called natural language processing (NLP). One of the most prominent area in NLP is sentimental analysis. Sentimental analysis in machine learning is usually applied on three levels, sentence level, document level, and aspect level [2]. Sentence level analyzes the sentiment on each sentence. Document level classifies the entire document as binary class or multi-class. Aspect level is a more complicated level, which is identifying the different aspects of a corpus first, and then classifying each document with respect to the observed aspects of each document.

The report aims to classify the sentimental representations of Internet Movie Database (IMDb) reviews via machine learning based classification on document level. The report will first remove the stop words and normalize words in the IMDb reviews to better the performance of the classification. In next step, the report will transform the reviews into the word matrix, which represents the features for the classification. Last, several algorithms (logistic regression, SVM, Naïve Bayes, random forest, boosting, deep neural networks) are utilized to train and test the word matrix to evaluate which algorithm can perform better on this classification.

The following report is organized as follows. Chapter Two presents the related work on sentimental analysis via machine learning. Chapter Three illustrates the methodology of this report. Chapter Four discusses the accuracy of this report. Chapter Five concludes the report and points out the future possible research direction.

## 2. Related work

Tripathy et al. [1] presented a text classification by using Naïve Bayes (NB) and support vector machine (SVM). The results showed that these two algorithms can classify the dataset with high accuracy compared to the other existing research.

Sharma et al. [3] classified the sentiment of the short sentences via convolutional neural network (CNN) with Word2Vec vectorization. The authors cleaned the data with Word2Vec, and implemented CNN to solve the issues of inconsistent noise in language. The results showed that CNN was able to extract better features for short sentences categorization.

Vijayaragavan et al. [4] discussed an optimal SVM based classification for the sentimental analysis of online product reviews. The paper firstly applied SVM and K-means to cluster the reviews into two groups. Then, the authors employed fuzzy based soft set theory to determine the possibility of customer to purchase the product.

However, the above research limited on the exploration of different algorithms to better the classification. The report will, therefore, extend the previous research's effort to more choices of algorithms for a better prediction accuracy.

## 3. Methodology

The report utilizes a methodology to conduct the analysis of the sentiment analysis of IMDb reviews, as shown in Fig. 1. First, the report illustrates and feeds the data into the data cleaning and preprocess. Next, the report removes the stop words and some irrelevant words from the original data; then, the vectorization techniques are applied to transform text into a feature matrix. Last, the report applies six different algorithms to train and test the feature matrix.

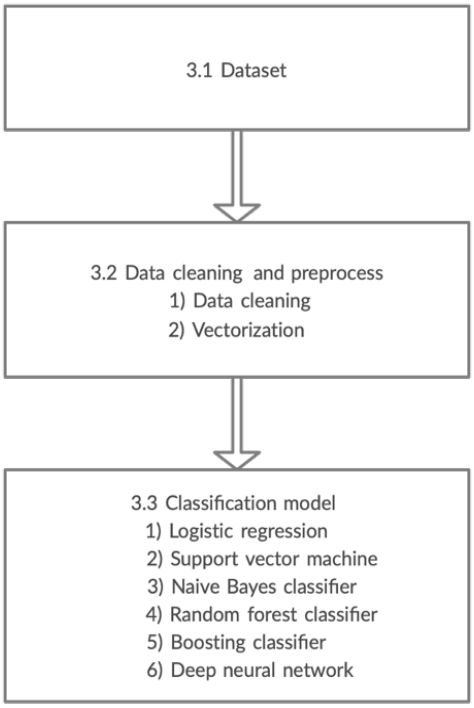


Fig. 1. methodology of the report

### 3.1 Dataset

The dataset is retrieved using the method described in [5, 6]. This dataset consists of 50,000 movie reviews taken from IMDb. Half of the data is used for training while the other half is used for testing. Moreover, both the training and testing dataset have 50% of positive reviews and 50% of negative reviews.

In each of the reviews, users are allowed to rate the movie from 1 to 10. In order to transform this rating scale to a binary label, we define a review as negative if its rating is less than 4 and positive if its rating is more than 7, reviews with ratings between this range are omitted.

### 3.2 Data cleaning and preprocess

#### 1) Data cleaning

In order to facilitate the data interpretation in the later work, raw texts obtained from the previous section are preprocessed. First, elements such as punctuations, line breaks, numbers, and stop words like 'a', 'the', and 'of' are removed since they provide little information about the user's impression towards a movie. Then, all the words are converted to lower cases and normalized to its true root (e.g. played to play) in order to reduce the noise of the vocabularies.

#### 2) Vectorization

Vectorization is the process of transforming the text data into numeric representations so that the data can be understandable by machine learning algorithms. In this project, we use 4 different methods of vectorization:

- Binary vectorization

One of the simplest vectorization methods is to represent the data as a binary-valued $n \times m$ matrix, where the element $i_{n,m} \in \{0,1\}$ denotes the existence of the $n^{th}$ vocabulary of the corpus in the $m^{th}$ movie review.

- Word-count vectorization

We can also replace the binary values in the matrix with word counts, in which the element of the matrix $i_{n,m} \in \mathbb{R}$ now becomes the frequency that the $n^{th}$ vocabulary of the corpus appears in the $m^{th}$ movie review. This method increases the weight of the more frequently-appeared words in the predictions.

- n-grams vectorization

In the vectorization method mentioned above, each column of the matrix represents a unique word in our corpus, which means that we are using the appearance of words as our features to predict the rating of a movie review. However, we can also expand the features to a group of consecutive words, called the n-grams of the texts. For example, if we are using n-grams of size 3 to vectorize our data, the columns of the matrix become a sequence of 3 consecutive words appeared in our corpus. This method is useful in cases when phrases provides more information for the prediction than individual word.

- tf-idf vectorization

The term frequency-inverse document frequency (tf-idf) is a measure of how a given word is concentrated into relatively few documents [11]. This method is based on the idea that the terms which appear more frequently and concentratedly in fewer documents are more representative of the content in the documents.

3.3 Classification model

The report implements six classification models to analyze the sentiment of the context, including logistic regression, support vector machine, Naïve Bayes classifier, random forest classifier, boosting classifier, and deep neural networks.

1) Logistic regression

Logistic regression performs the binary classification by using a sigmoid function as the hypothesis, which is given by:

$$P(y = 1|x; \theta) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

The logistic regression model is trained by fitting the parameter $\theta$ via maximum likelihood, where the log likelihood function can be represented as:

$$\ell(\theta) = \sum_{i=1}^{n} y^{(i)} \log h\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h\left(x^{(i)}\right)\right)$$

Then, $\theta$ can be updated using stochastic gradient ascent rule

$$\theta_j := \theta_j + \alpha \frac{\partial}{\partial \theta_j} \ell(\theta)$$

$$:= \theta_j + \alpha \left(y^{(i)} - h_\theta\left(x^{(i)}\right)\right) x_j^{(i)}$$

The setting of logistic regression in this report:

- Inverse of regularization value: [0.01, 0.05, 0.25, 0.5, 1], choose the best performance
- Penalty = l2
- Tolerance = 1e-4

2) Support vector machine

Support vector machine (SVM) is considered as one of the best algorithms for supervised learning. The main idea of this algorithm is to map the data from a relatively low dimensional space to a relatively high dimensional space so that the higher dimensional data can be separated into two classes by a hyper plane. The hyperplane that separates the data with maximum margin is called the support vector classifier, which can be determined using Kernel Functions in order to avoid expensive computation to transform the data explicitly [10].

The setting of SVM in this report:

- Inverse of regularization value: [0.01, 0.05, 0.25, 0.5, 1], choose the best performance
- Penalty = l2
- Tolerance = 1e-4

3) Naïve Bayes classifier

The Multinomial Naïve Bayes algorithm is useful in the case when the features $x_j$ are discrete-valued due to its simplicity and ease of implementation. This algorithm is based on a strong assumption that $x_i$'s is conditionally independent given $y$, which is also known as the Naïve Bayes (NB) assumption [10]. The model is parameterized by $\phi_{j|y=1}$, $\phi_{j|y=0}$, and $\phi_y$, these parameters can be calculated as:

$$\phi_{j|y=1} = p\left(x_j = 1 \middle| y = 1\right) = \frac{\sum_{i=1}^{n} 1\left\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\right\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}}$$

$$\phi_{j|y=0} = p(x_j = 1|y = 0) = \frac{\sum_{i=1}^{n} 1\left\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\right\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 0\}}$$

$$\phi_y = p(y = 1) = \frac{\sum_{i=1}^{n} 1\left\{y^{(i)} = 1\right\}}{n}$$

After fitting the parameters, the prediction on a new sample with features $x$ can be obtained as:

$$p(y = 1|x) = \frac{\left(\prod_{j=1}^{d} p\left(x_j \middle| y = 1\right)\right) p(y = 1)}{\left(\prod_{j=1}^{d} p\left(x_j \middle| y = 1\right)\right) p(y = 1) + \left(\prod_{j=1}^{d} p\left(x_j \middle| y = 0\right)\right) p(y = 0)}$$

The setting of Naïve Bayes classifier in this report:

- Laplace smoothing: 1

4) Random forest classifier

Tree classification is very powerful to classify the nonlinear dataset, like NLP. The classification includes bagged tree, random forest, and boosting [8]. Random forest provides an improvement over the bagged trees. Bagged trees consider all the predictors (p predictors) in every split of the tree, whereas

random forest limits the selection of the predictors to m predictors. The number of predictors considered in the split in random forest is equal to the square root of the total number of predictors, $m = \sqrt{p}$. In other words, random forest decorrelates the trees through considering less predictors. Unlike highly correlated bagged trees, the variance in random forest is significantly decreased [8].

The setting of random forest in this report:

- The number of trees: 100
- Quality criterion: Gini index.
  K is the class number. M is the sample size. The value will take on a small value if the node is pure.

$$G = \sum_{K=1}^{K} p_{mk} \, log \, p_{mk}$$

- The maximum depth of the tree: None
- The minimum number of samples required to split an internal node: 2

5) Boosting classifier

Boosting classifier is another approach of tree classification. Boosting also becomes a method to improve the predictions over bagged trees. Boosting trees are grown sequentially. Each tree is grown based on the information from previously grown trees, thus robust to overfitting. Notably, boosting does not involve bootstrap sampling; instead each tree collectively fit on the original tree [8].

The setting of boosting in this report:

- The number of boosting trees: 100
- Test criterion: MSE.
- Learning rate: 0.1
- The minimum number of samples required to split an internal node: 2

6) Deep neural networks (DNN)

Neural network is recognized as a useful tool for nonlinear statistical modeling [9]. This model is able to incorporate combinations of different neurons (functions) into one giant network [10]. Neural network has evolved to encompass a large class of models and learning algorithms, such as deep neural networks, convolutional neural networks, recurrent neural networks. This report utilizes a five-layer deep neural networks to classify the sentiment of the language.

The setting of DNN in this report:

- The hidden layer: (30, 30, 20, 10, 10)
- Activation function for the hidden layer: Logistic function
- L2 penalty (regularization term): 0.0001
- Early stopping: True
- The solver for weight optimization: Adam

## 4. Results and discussion

The report evaluates the algorithms' performance by the confusion matrix. A confusion matrix shows the relation between the correct and wrong predictions, as shown in Table. 1.

Table. 1. confusion matrix

| | | True Labels | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted Labels | Positive | True Positive (TP) | False Positive (FP) |
| | Negative | False Negative (FN) | True Negative (TN) |

The matrix provides several evaluation parameters, including:

- Positive precision: the accuracy of the positive prediction.

$$Positive\ precision = \frac{TP}{TP + FP}$$

- Negative precision: the accuracy of the negative prediction.

$$Negative\ precision = \frac{TN}{TN + FN}$$

- Accuracy: the ratio of the correct predictions, which is the average of negative and positive precision.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Table. 2 shows the results of the evaluation parameters of confusion matrix for each algorithm.

Table. 2 the table of each algorithm performance

| Algorithm | Vectorization | Regularization | Positive precision | Negative precision | Accuracy |
|---|---|---|---|---|---|
| Logistic | binary, 3 grams | 1 | 0.908 | 0.893 | 0.900 |
| | word count, 3 grams | 1 | 0.899 | 0.894 | 0.897 |
| | tf-idf, 3 grams | 1 | 0.881 | 0.872 | 0.877 |
| SVM | binary, 3 grams | 100 | 0.908 | 0.894 | 0.901 |
| | word count, 3 grams | 20 | 0.900 | 0.895 | 0.898 |
| | tf-idf, 3 grams | 1 | 0.904 | 0.896 | 0.900 |
| Naïve Bayes classifier | binary, 3 grams | - | 0.839 | 0.923 | 0.881 |
| | word count, 3 grams | - | 0.836 | 0.912 | 0.874 |
| | tf-idf, 3 grams | - | 0.819 | 0.868 | 0.879 |
| Random Forest classifier | binary, 3 grams | - | 0.859 | 0.845 | 0.852 |
| | word count, 3 grams | - | 0.860 | 0.839 | 0.849 |
| | tf-idf, 3 grams | - | 0.864 | 0.786 | 0.844 |
| Boosting classifier | binary, 3 grams | - | 0.863 | 0.798 | 0.831 |
| | word count, 3 grams | - | 0.869 | 0.800 | 0.834 |
| | tf-idf, 3 grams | - | 0.865 | 0.789 | 0.825 |
| Deep neural network | binary, 3 grams | 0.0001 | 0.911 | 0.901 | 0.906 |
| | word count, 3 grams | 0.0001 | 0.896 | 0.900 | 0.898 |
| | tf-idf, 3 grams | 0.0001 | 0.881 | 0.921 | 0.901 |

- Vectorization

As the table shows, the binary and 3 grams vectorization perform best among all three vectorizations for all the algorithms. The reason for this may be binary vectorization can reduce the noise of the parameters. The word-count and tf-idf vectorization count the number of the word in the matrix. In this case, some irrelevant words are counted multiple times, increasing the variance of the model.

- Regularization

As for regularization, SVM has many noise parameters since it has a low inverse of the regularization value, whereas logistic regression and DNN fits a model with a relatively low regularization strength. One possible way to further improve SVM is trying to increase the regularization value, minimizing the noise parameters as much as possible. Or the other method may be clean the data more comprehensively, for example, removing subject term should be helpful for the prediction, since the adjective term contributes more to the accurate predictions.

- Positive and negative precision

These two estimates provide us a tool to evaluate the accuracy of positive and negative predictions. Logistic regression, SVM, Random Forest, and boosting contribute to a better positive prediction accuracy, whereas Naïve Bayes classifier and DNN contribute to a better negative prediction accuracy, about 92% (highest among all the models). In other words, if the scenarios need more accurate negative predictions, the users can implement Naïve Bayes classifier and DNN to predict, whereas if the positive predictions matter more, then the users can implement Logistic regression, SVM, random forest, and boosting.

- Accuracy

In term of accuracy, DNN in binary and 3 grams vectorization performs best, 90.6% accuracy. Five hidden layers of DNN are able to account for more non-linear relationship of the dataset. It won't be surprised if DNN with more layers can provide better results. In addition, logistic regression and SVM in binary and 3 grams vectorization also perform well (90%) in shorter amount of time. Especially surprising is that Naïve Bayes in binary and 3 grams vectorization

has 88% accuracy. It is the easiest model among all six models, meaning further improving the performance of predictions is possible.

## 5. Conclusion and future work

The report proposes a methodology to conduct the sentiment analysis of IMDb reviews. The methodology has three major steps, as shown in Fig. 1. As the results show, the binary and 3 grams vectorization performs best among all three vectorizations for all the algorithms. In term of negative accuracy, Naïve Bayes classifier and DNN contribute to a better prediction, whereas the other four models perform better on the positive prediction. In addition, DNN, logistic regression, and SVM provide 90% prediction accuracy, which is very promising for the sentiment analysis

Last, future work should implement other vectorizations to better the word matrix. For instance, the researchers can try to remove the subjects in the sentences. In addition, future work can try more complicated models for the analysis. For example, recurrent neural network may be able to provide better performance since it is able to further account for the relationship of the sentences.

## GitHub code

https://github.com/JoshWuuu/cs229-final-project.git

## Reference


[1] A. Tripathy, A. Agrawal, and S. K. Rath, "Classification of Sentimental Reviews Using Machine Learning Techniques", 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015), Procedia Computer Science, vol. 57, 2015, pp. 821 – 829.

[2] R. Feldman, "Techniques and applications for sentiment analysis," Communications of the ACM, vol. 56, 2013, pp. 82–89.

[3] A. K. Sharma, S. Chaurasia, and D. K. Srivastava, "Sentimental Short Sentences Classification by Using CNN Deep Learning Model with Fine Tuned Word2Vec", International Conference on Computational Intelligence and Data Science (ICCIDS 2019), Procedia Computer Science vol. 167, 2020, 1139–1147.

[4] P. Vijayaragavan, R. Ponnusamy, and M. Aramudhan, "An optimal support vector machine based classification model for sentimental analysis of online product reviews", Future Generation Computer Systems, vol. 111, 2020, 234–240.

[5] A. Kub, "Sentiment Analysis with Python (Part1)", https://towardsdatascience.com/sentiment-analysis-with-python-part-1-5ce197074184, accessed on Jun. 5, 2020.

[6] A. Kub, "Sentiment Analysis with Python (Part2)", https://towardsdatascience.com/sentiment-analysis-with-python-part-2-4f71e7bde59a, accessed on Jun. 5, 2020.

[7] S. Bansal, "A Comprehensive Guide to Understand and Implement Text Classification in Python", https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/, accessed on Jun. 5, 2020.

[8] G. James, D. Witten, T. Hastie, and R. Tibshirani, "An Introduction to Statistical Learning: with Applications in R", Springer Publishing Company, Incorporated, 2014.

[9] T. Hastie, R. Tibshirani, J. H. Friedman, "The elements of statistical learning: data mining, inference, and prediction. 2nd ed", New York: Springer, 2009.

[10] M. Tengyu, A. Avati, K. Katanforoosh, and A. Ng, "CS 229 machine learning", class handout, Stanford University, 2020.

[11] J. Leskovec, A. Rajaraman, and J. D. Ullman. "Mining of Massive Datasets (2nd. ed.)", Cambridge University Press, USA, Chapter 1, pp. 8-19, 2014.