Sentiment Analysis on Textual Datasets using Superior Classifiers

Mohammad Amini, Renato Salinas, Zainab Almheiri

***Abstract*—Sentiment analysis is a common task in natural language prepossessing (NLP) that applies to textual datasets to classify and interpret emotions into positive and negative labels. This application is useful in real life application. For example in businesses and more specific examples such as the second dataset, IMDB movie reviews, sentiment analysis can be applied to identify customers emotions towards products, and movies (negative or positive), respectively. In this study, we apply different classifiers on IMDB movie reviews to predict the emotions to positive and negative and also on a 20 news groups to predict that a certain text belongs to a certain news group. The classifiers used to try to train and predict those two possible outcomes are Logistic Regression, SVM, Decicion Trees, Adaboost, Random Trees, Multinomial Naive Bayes, and Naive Bayes SVMs. After applying simple prepossessing to maintain the model performance and using a vectorizer to tokenize the words, our results after applying the models shows that Naive Bayes SVM was the best one at predicting the IMBd data set, while the Multinomial Naive Bayes was the best accuracy when predicting the 20 news group dataset.**

***Keywords*—Sentiment Analysis, Textual Classification, Applied Models,**

I. Introduction

IMDB is a popular website that consists of record information related to television programs, games, home videos, and movies. Predictive models can be built using this database paired up with sentiment analysis to classify reviewer opinions as either positive or negative. Such approach is used with a dataset from IMDB reviews. In parallel, 20newsgroup is a popular dataset that can be downloaded by the sklearn, in which news are separated in 20 different topics. Predictive models can be built using this dataset paired up with tokenization and information retrieval for short term frequency to classify news as either one of the 20 topics. The approach we followed was to build predictive models such as SVM, Decision trees, Ada boost, logistic regression, random forest, multinomial Naive Bayes, and Naive Bayes SVMs. The models were then tuned and optimized. Then we compared the performance of such models for both datasets.

In order to build an accurate model, we conduct preprocessing on both dataset. We conduct simple preprocessing on IMDB and 20 news group datasets because as we complicate the preprocessing the model accuracy/precision decreases. We also used tokenization using information retrieval for short term frequency to have the most information out of our words, giving less importance for words that would not help with the learning, such as determiners.

The results prove that as the IMDb dataset is binary, it was easier for our models to get a higher accuracy than with the 20newsgroup dataset which is multiclass. We managed to achieve the best performance on 20newsgroup dataset thanks to multinomial Naive Bayes, with 0.70 expected accuracy. As for the IMDb, we achieved the best performance with Naive Bayes SVM, with 0.90 expected accuracy.

The rest of the paper is organized as follows: Section II summarizes Related Work, Section III discusses Datasets and Model setup including preprocessing and feature extraction. Section IV contains Mathematical representation of the applied models and their description. Section V includes the Proposed Methods which are extra methods that we tried. Section VI elaborates on Model Implementation which includes the feature representation. Section VII summarizes the results with the experiment setting. We conclude the paper in section VIII. Finally, in section IX, we explain the contribution of the teammates.

II. Related Work

Sentiment analysis is a set of techniques, methods, and tools that aim to extract and detect subjective information from textual data (language) such as attitudes and opinions. In real-life applications such as stock markets, software engineering, elections, disasters, and medicine sentiment analysis is very common. Especially when customer's opinions (negative, positive or neutral) matters [1]. Liu [2] conducted a comprehensive review of the statistical learning methods that can be applied to binary and multi-classification.

III. Datasets and preprocessing

In this study, we use two datasets: IMDB and 20 news group datasets. We apply preprocessing method which is an essential step to maintain high model performance. This section discuses the applied datasets, our chosen preprocessing method and feature extraction.

A. IMDB Reviews Dataset

The data consists of 25000 training and 25000 testing reviews. Additionally, the data contains 12500 positive (rating of 7 or greater) and 12500 negative (rating of 4 or less) reviews in the training and test sets. Thus, the data is balanced.

B. 20 News Group Dataset

The data consists of approximately 18000 newsgroups posts on 20 topics which are split into training and testing. How the split is done is based upon messages posted before and after a specific date.

C. Preprocessing

The data needed to be preprocessed in order to avoid feeding the algorithms any data, words, that might not be useful for learning, while also allowing the algorithms to be able to learn from that information. To do so, we used mainly three approaches: we cleaned the strings from punctuations, turned everything into lowercase, and removed spaces, hyphens and other regex.

First, we removed spaces, hyphens, and other regex. Then, we removed any punctuation from the sentences as they were seen unnecessary. Afterwards, we turned the new string into lowercase. All of this was done to avoid any confusion from the algorithm since we are focusing on words and random regex, symbols and malformed duplicates written differently could otherwise hinder the training of the models.

Additionally, to deal with some social media writing such as IMDB reviews data set, for AdaBoost and Random Forest, we applied some text normalization techniques as following: 1. Remove punctuations. 2. Replace all contractors with the formal forms. 3. Create a Regex path to replace numbers, times and other non-words to the specific names. 4. Use negation detection process, because of the nature of this assignment which is sentiment analysis related. 5. Correct any misspelling.

D. Feature Extraction

To deal with the training corpus of the IMDB reviews and 20 news group, in term of this section we search and compare some of techniques and based of previous researches [3] there are some well-known algorithms (i.e word2vec, TF-IDF, etc.).

We tokenized the data by using a Tf-Idf vectorizer in order to make sure that words that appear in the data with too much frequency would not have too much impact, since such words usually don't give much interesting information, such as determiners, which would not help much with the learning process. Afterwards, in some cases, we normalized the data to make it easier for some methods to approach, such as logistic regression and multinomial Naive Bayes.

To make a comparison between word2vec and TF-IDF, Adaboost and Random Forest have been deployed with both techniques on IMDB reviews data set. In order to build the word2vec model we tuned the parameters and an example of the output model is shown on Table I:

TABLE I: The most 5 word similar for word "school" with their cosine similarity

word	class	high_school	college	teacher	student
cosine similarity	0.712	0.638	0.633	0.618	0.580

For TF-IDF we used unigram and bigram with word analyzer parameters. The comparison of both techniqus using AdaBoost and Random Forest models on IMDB reviews validation set are showed in Table II:

TABLE II: Performance comparison for TF-IDF and word2Vec.

Vectorizer	Random Forest Accuracy	AdaBoost Accuracy
word2vec	0.82	0.80
TF-IDF	0.85	0.61

IV. Mathematical Representation and Setup of the Applied Models

A. Support Vector Machines (SVMs)

Support vector machines (SVMs) are supervised learning models that can apply for classification and regression. Given a set of labeled data, SVMs training algorithms builds a model that assign new labels to one category. In this study, we develop SVMs for binary and multi-classification analysis. Although we use SVMs to predict binary and multi-classification, in this section we will represent the mathematical formulation of SVMs solving binary classification. In binary classification using linear models formulation:

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}) + b \tag{1}$$

Where $phi(\mathbf{x})$ refers to feature space transformation, and b is the bias term. Multiple optimal solutions can be found, however, we should choose the one that provide the smallest generalization error. The SVMs approaches this by finding the maximum margin which is defined to be the smallest distance between the decision boundary and any of the observations. For example, Eq. (2) represents the constraints to find the decision boundary with maximum hard margin.

$$\begin{cases} \max_{w,w_0} M \\ M \leq \frac{1}{\|w\|_2} y^{(n)} \left(w^\top x^{(n)} + w_0\right) \quad \forall n \end{cases} \tag{2}$$

SVMs can efficiently performs both linear and non-linear classification [4]. In SVMs, the hyperparameters are the type of the kernel function, C and gamma (based on the kernel function). For 20newsgroup dataset, two optimal solutions are obtained.The first optimal solution is achieved at: kernel= linear function, C= 10. Besides, the second optimal solution is reached at: kernel= radical basis function (RBF), C= 1000, and gamma = 0.001. However, the second optimal solution is reported as it achieved a bit higher accuracy the optiaml solution 1 (see results section). In addition, the optimal solution for IMDB dataset is achieved when the kernel function is radical, C=1, and gammas = 0.001. Result section presents the detail performance of each SVMs applied on both binary and multi-classification.

B. Decision Trees

In decision tree, we divide input space into regions and learn one function per region that learned adaptively:

$$f(x) = \sum_k w_k \mathbb{I}\left(x \in \mathbb{R}_k\right) \tag{3}$$

Regions are successively splitted based on a set of condition, and based on the value of a single variable called test. In classification, we count the frequency of classes per each reation and then predict the probability of the most frequent label:

$$w_k = \mathrm{mode}\left(y^{(n)} | x^{(n)} \in \mathbb{R}_k\right) \tag{4}$$

Tuning hyper-parameters is a paramount step in all learning algorithms. Decision trees have many hyper parameters such as the function to measure the quality of a split (gini or entropy),

the maximum depth of the tree (max-depth), the minimum sample required to split an internal node (min-samples-leaf), the number of feature to be considered when looking for the best split (max-features), and maximum leaf nodes (max-leaf-nodes). We tune decision trees based on gini criterion, (min-samples-leaf), (max-features), and (max-leaf-nodes). For 20newsgroup dataset, after tuning the hyper-parameter under different range of min-samples-leaf, max-leaf-nodes, and max-features. The optimal solution is achieved at: max-features = 60000, max-leaf-nodes = 9000, and min-samples-leaf=1. For IMDB reviews dataset, the optimal solution is accomplished at: max-depth = 18, max-features= 80000, max-leaf-nodes= 3000, and min-sample-leaf=8 (see results section for accuracy and precision).

C. Logistic Regression

Logistic Regression, also known as maximum-entropy classification, is a classifier used to model the probability of a certain event existing such as pass/fail (single trial). This training algorithm was used with a one-vs-rest approach for the IMDb data and a cross-entropy loss approach for the 20newsgroup data. By using such logistic regression in sklearn, regularization is applied by default as it can handle dense and sparse input. We adapted to L2 regularization for both datasets as it was shown to be the best between L1 and L2.

Due to regularization, the alpha, learning rate, needed to be a bigger number than what we would've used without regularization, as the alpha is usually expected to be around 0.01. This is especially shown in the 20newsgroups dataset in Table III, in which we get a learning rate of 100 as the optimal number with the help of GridSearchCV with a CV of 10. As shown in table III, we see how the alpha changes, particularly increasing the learning rate, increases the accuracy that we get, up to a cap, in which in some cases, such as in the table VII, the accuracy starts to drop:

TABLE III: Performance of Logistic Regression models.

Learning Rate	20newsgroup Accuracy	IMDb Accuracy
0.001	0.055	0.769
0.010	0.359	0.784
0.100	0.591	0.834
1.000	0.712	0.865
10.00	0.733	0.859
100.0	0.733	0.847
1000.0	0.733	0.841

Since convergence is necessary to get accurate results, the numbers of iterations needed to be increased. After optimizing the parameters mentioned above, the optimal solution for datasets was found so far by the following settings:

	20newsgroup	IMDb
Alpha	100.0	1.0
Solver	Stochastic Average Gradient	Stochastic Average Gradient
# of Iterations	4000	1000
Regularization	L2	L2

The objective of regression is to minimize the sum of squared residuals as seen in (5). When trying to minimize the square error loss (also known as L2 loss), we are required to be able to find the optimal points, in which a gradient descent tries to approach by finding a local min. It would require way too many iterations if we were to follow a regular gradient descent. To fix this issue, Stochastic Gradient Descent (SGD) picks a random data point from the whole dataset at each iteration, which in the end reduces the computation needed.

$$w^* = \arg\min_w \frac{1}{2} \sum_{n=1}^{N} \left(w^T x^{(n)} - y^{(n)} \right)^2 \tag{5}$$

D. Random Forest

Random forest, as its name implies, is made up of a large number of individual decision trees which act as an ensemble. Feature importance is one of ensemble techniques that implies, the value of the function is measured as the decrease in impurity of nodes weighted by the likelihood of hitting the node. The probability of nodes may be calculated by the number of samples reaching the node, divided by the total number of samples. The higher the value, the more that feature is relevant.To implement in Scikit-learn, each decision tree calculates a nodes importance using Gini Importance, assuming only two child nodes:

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)} \tag{6}$$

Then, the importance for each feature is calulate as:

$$fi_i = \frac{\sum_{j:\text{node j splits on feature i}} ni_j}{\sum_{k \in \text{all nodes}} ni_k} \tag{7}$$

The sum of the feature's importance value on each trees is calculated and divided by the total number of trees:

$$RFfi_i = \frac{\sum_{j \in \text{all trees}} \text{norm } fi_{ij}}{T} \tag{8}$$

Due to the number of features to consider when looking for the best split by using Random Search Training to pick the best parameters and make prediction by using GridSearchCV with cv of 5, we looking for optimal feature importance as seen on table IV:

TABLE IV: Performance of Random Forest model by modifying feature importance.

max_features	20newsgroup Accuracy	IMDb Accuracy
log2	0.571	0.763
sqrt	0.583	0.778

According to best hyper-parameters selection by Random Search Training which are bootstrap, max_depth, max_features, min_samples split, and n_estimators, with cv = 5, the performance in each cv for both 20newsgroup and IMDB reviews has been showed in Figure 1.

The high influentially parameters for our Data sets are n_estimators and max_features.

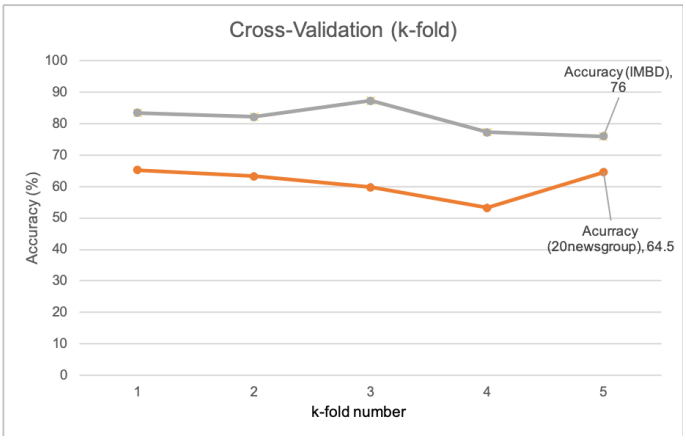

Fig. 1: Performance of Random Forest model

E. Adaboost

The AdaBoost algorithm, introduced in 1995 by Freund and Schapire [5]. One of the algorithm's main ideas is to establish a distribution or collection of weights over the training set. The weight of this distribution on training example i on round t is denoted by Dt. Initially, all weights are set equal, but each round raises the weights of incorrect categorized examples, so that the weak learner is forced to focus on the hard examples in the training set [6]. The weak learner's job is to find a weak hypothesis ht appropriate for the distribution Dt.The goodness of a weak hypothesis is measured by its error:

$$\epsilon_t = \Pr_{i \sim D_t} \left[h_t \left(x_i \right) \neq y_i \right] = \sum_{i : h_t(x_i) \neq y_i} D_t(i) \qquad (9)$$

To implement AdaBoost algorithm is implemented in Scikit-learn library that we used for Both Data sets. There are five parameters in this classifier which choosing some of them is clear and no need to use best model selection techniques like parameter algorithm including SAMME and SAMME.R, that based on the explanation the SAMME.R algorithm generally converges faster than SAMME, and has lower test error with lower boosting iterations. Therefore, to chose best hyper-parameters using GridSearchCV we used n_estimators and learning_rate to find the best hyper-parameters. the performance of the model with fix number of 1000 estimator and four different learning rate is shown on Table V:

TABLE V: Performance of AdaBoost model with different learning rate.

Learning Rate	20newsgroup Accuracy	IMDb Accuracy
0.01	0.273	0.742
0.05	0.321	0.753
0.1	0.380	0.769
0.3	0.429	0.785

The high influentially parameter for our Data sets is learning rate.

V. Proposed Methods

This section presents two additional applied algorithms called NBSVM, and Multinomial NB.

A. Multinomial NB

Multinomial Naive is a generative classifier that is suitable for classification of discrete features such as word counts for text classification, as such, it was chosen as one of the extra classifiers we used for the classification of textual data. Such distribution of the data is parametrized by vectors $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ for each label, where the number of features is the size of the vocabulary and θ_{yi} is the probability of feature i appearing in a sample that belongs to a certain label. The parameters are then estimated by relative frequency counting as shown in (10).

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n} \qquad (10)$$

As seen in (10), the tuning of the parameter alpha is crucial for try to get the best approach. With the help of GridSearchCV with a cv of 10, we managed to find an optimal alpha as seen on table VI.

TABLE VI: Performance of Logistic Regression models.

Alpha	20newsgroup Accuracy	IMDb Accuracy
0.001	0.743	0.696
0.010	0.752	0.730
0.100	0.728	0.773
1.000	0.648	0.807
10.00	0.463	0.806
100.0	0.303	0.747
1000.0	0.055	0.720

Given table VI, the optimal parameters for the datasets given Multinomial Naive Bayes are 0.01 for 20newsgroup and 1.0 for IMDb. According to such parameters, in the case of the 20newsgroup dataset, we use Lidstone smoothing and for the IMDb dataset, we use Laplace smoothing

B. Naive Bayes SVMs (NBSVM)

In this study, we apply NBSVM method that is proposed by [7]. This model is a mix of unsupervised and supervised learning methods to capture semantic term. The results proved that the proposed model is robust (see [7] for more detail). This model can be applied for binary classification analysis. Thus, we apply NBSVM to IMDB reviews detest. This model has two hyperparameters including C and beta. Although, we apply RBF for IMDB reviews dataset. In, NBSVM we apply linear kernel and obtain approximately same accuracy as for SVMs with kernel function (see result section). Tabel 1. depicts the performance of NBSVM at different C and beta. The highest accuracy is achieved at C= 1 and beta = 0.25.

TABLE VII: Performance of NBSVM model.

Accuracy	beta= 0.25	beta=0.5	beta=0.75	beta=0.95
C=1	**0.873**	0.871	0.87	0.826
C=2	0.869	0.87	0.865	0.816
C=3	0.869	0.87	0.861	0.809
C=4	0.864	0.869	0.857	0.805

VI. Model Implementation

A. Feature Representation

Let d^i be the feature vector for ith review and y^i be the label of each ith observation. In newsgroup dataset, there are 20 labels from 1 to 20. In IMDB reviews dataset, we label the outputs as follow:

$$y^i = \begin{cases} 1, & \text{if positive review} \\ 0, & \text{if negative review} \end{cases} \quad (11)$$

VII. Results

A. Experiment Setting

Jupyter notebook is used as running environment to implement all the models. Three different RAM and CPU are used. Two methods of Cross validation are applied including hold-out and k-fold.

B. Result of the Final (selected) Models

Table VIII and Tabel IX depict the performance of all applied models including SVMs, decision trees, logistic regression, AdaBoost, and random forest. Besides, the proposed model, multinomialNB and NBSVM on newsgroup and IMDB reviews. As previously mentioned, NBSVM is applied to the IMDB dataset (binary classification). The results prove that logistic regression, MultinomialNB, and SVMs achieved the highest accuracy on the 20newgroups dataset (see Table VIII). Whereas, MultinomialNB and NBSVM achieved the highest accuracy to the IMDB dataset, 0.88 and 0.90, respectively (see Table IX). Although decision trees are the least in terms of computational cost, it performed better on binary than multiclassification (see Table VIII and IX). Additionally, logistic regression can be superior to both binary and multi-classification.

TABLE VIII: Performance of the Applied and Proposed models on 20newsgroup Dataset.

Model	Accuracy	TrainingTime (sec)	PredictionTime (sec)
Logistic Regression	**0.66**	855.665	0.048
Decision Trees	0.34	23.462	0.012
SVMs	**0.64**	78.753	39.176
Adaboost	0.42	12240	7.485
Random Forest	0.61	124743	15.921
Multinomial NB	**0.70***	459.855	0.041

VIII. Conclusion and Future Recommendations

In this study, we applied powerful yet basic machine learning classifiers. The results prove the superiority of logistic regression, multinomialNB, and SVMs to both binary and multi-classification analysis. MultinomialNB achieved the highest accuracy (0.70) on the 20newsgroup. Whereas, NBSVM achieved the highest accuracy (0.90) on IMDB dataset. It is worth to mention that SVMs, decision trees, logistic regression, and multinomial are efficient classifiers in terms of computational training and prediction time. NBSVM is the most efficient algorthims to IMDB dataset in terms on computational cost and accuracy. NBSVM also showed to be the most efficient when it came to time for training and prediction.

TABLE IX: Performance of the Applied and Proposed models on IMDB reviews Dataset.

Model	Accuracy	Trainingtime (sec)	Predictiontime (sec)
Logistic Regression	**0.88**	263.71	0.012
Decision Trees	0.72	16.647	0.041
SVMs	**0.88**	451.324	404.358
Adaboost	0.78	5881	5.42
Random Forest	0.79	67503	12.01
NBSVM	**0.90***	1.05	0.169
Multinomial NB	**0.83**	79.099	0.026

For future recommendations, we suggest improving the NBSVM classifier in order to apply to multiclassification analysis as it is a very powerful classifiers. In addition, we suggest trying Artificial neural networks (deep learning) on both datasets as they would definitely give better accuracy. Finally, we suggest conducting further preprocessing on the 20newsgroup dataset and make sure that the data is balanced.

IX. Statement of Contribution

We all contributed in the division of the preprocessing and feature extraction. For the models, we divided the models equally and helped each other whenever we had questions during our weekly meetings and online. We all contributed in the report and putting things together for the evaluation.

References

[1] M. V. Mäntylä, D. Graziotin, and M. Kuutila, "The evolution of sentiment analysis—a review of research topics, venues, and top cited papers," *Computer Science Review*, vol. 27, pp. 16–32, 2018.

[2] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[3] R. Dzisevič and D. Šešok, "Text classification using different feature extraction approaches," in *2019 Open Conference of Electrical, Electronic and Information Sciences (eStream)*, pp. 1–4, IEEE, 2019.

[4] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[5] Y. Freund and R. E. Schapire, "Game theory, on-line prediction and boosting," in *Proceedings of the ninth annual conference on Computational learning theory*, pp. 325–332, 1996.

[6] Y. Freund, R. Schapire, and N. Abe, "A short introduction to boosting," *Journal-Japanese Society For Artificial Intelligence*, vol. 14, no. 771-780, p. 1612, 1999.

[7] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pp. 142–150, Association for Computational Linguistics, 2011.