Text Classification

Jingxu (Jack) Hu
jingxu.hu@mail.mcgill.ca

Alex Goulet
alex.goulet@mail.mcgill.ca

Karanvir Sidhu
karanvir.sidhu@mail.mcgill.ca

Abstract

This paper assesses the performance of a variety of machine learning classifiers used for text classification, with an emphasis on the Bernoulli Naïve Bayes classifier. The training and test data sets consist of posts or comments obtained directly from the Reddit website. Each post or comment can be categorized under one of eight possible class labels. For each of the classifiers, model selection was made using 5-Fold Cross-Validation. The results suggest that the final model accuracy on the test set depends mainly upon the various data preprocessing techniques, as well as the nature of the classifier that is used. More specifically, altering the data preprocessing pipeline and classifier parameters caused significant changes in accuracy for any given model. As such, the GridSearchCV method from the *scikit-learn* library was used to find the optimal combination of pipeline and classifier parameters. Furthermore, the results suggest that the model accuracy generally increases when the data has undergone both TF-IDF normalization and feature selection using the χ^2 test. Additionally, model accuracy increased, even more, when a small Laplace smoothing factor was applied to both the Bernoulli and Multinomial Naïve Bayes classifiers. As for LinearSVC, it was determined that the default parameters produced the best accuracy. The final model accuracy on the test set for Ensemble Bernoulli NB Bagging, Multinomial NB, Bernoulli NB, and LinearSVC were 88.50%, 88.80%, 91.139%, and 89.01%, respectively.

1 Introduction

1.1 Overview

The applications of machine learning are virtually endless, with countless new ways being discovered every day in an attempt to solve complex problems. Perhaps one of the most traditional and well-known applications of machine learning is text classification. In-text classification problems, words, or even groups of words become inputs to an algorithm that attempts to determine the context under which these words operate. To some extent, it could be interpreted as a way for computers to "understand" the meaning behind sentences. The task of acquiring a plethora of data on various topics is difficult. Reddit, a popular website where users post about any topic they so choose, is a good starting search area. The innumerable amount of posts or comments are regrouped under subreddits, which represent communities in which there exists an overarching theme. Consequently, this aspect of Reddit makes it the perfect candidate for the evaluation of text classification algorithms. This is because large quantities of input data with corresponding class labels are provided. The main purpose of this paper is to evaluate the performance of various text classification algorithms by using data obtained from different subreddits with an emphasis on the Bernoulli Naïve Bayes classifier. Data preprocessing and feature selection techniques directly influence the performance of text classifiers and will, therefore, also be explored. Consequently, over ten thousand posts or comments from 8 different subreddits will be used to train each text classifier so that afterward, they may be evaluated on the test set.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

1.2 Other Classifiers and Findings

Throughout the conducted experiments, in addition to the Bernoulli Naïve Bayes (NB) classifier, multiple other distinct classifiers were used in the prediction process. These classifiers were directly provided by the *Scikit-Learn* library and consist of Linear Support Vector Machine (LinearSVC), Ensemble Bagging with Bernoulli Naïve Bayes, and Multinomial Naïve Bayes (Multinomial NB). In order to compare the best possible accuracy between each classifier, an optimal data preprocessing pipeline was used for all of them. The optimal pipeline hyper-parameters were determined by using *Scikit-Learn*'s GridSearchCV function. The accuracy of the classifiers was obtained using their respective K-Fold Cross-Validation scores and turned out to be similar across all chosen classifiers. The final model accuracy on the test set for Ensemble Bagging with Bernoulli NB, Multinomial NB, Bernoulli NB, and LinearSVC was 88.50%, 88.80%, 91.139%, and 89.01% respectively.

2 Data Sets

2.1 Overview

The machine learning algorithms are used to solve a multiclass text classification problem, namely subreddit identification (subreddits are themed communities within the Reddit website). The training data set is composed of 11582 posts or comments that belong to one of the eight possible subreddits (classes), which are: "rpg", "anime", "datascience", "hardware", "cars", "gamernews", "gamedev", and "computers". As for the test set, it contains 2896 posts or comments which are to be mapped to one of the above subreddits by an algorithm. In order to do so, individual words are treated as features, and only the most relevant ones take part in training the different algorithms.

2.2 Feature analysis

Both the training and test data sets undergo multiple layers of preprocessing before their posts or comments can ultimately be transformed into feature vectors. The very first step is to remove all punctuation, capitalization, and spaces in order to obtain lists of lowercase words. The second step is to remove all the irrelevant words such as words with less than three letters, articles ("a", "the", "this", etc.), uninformative words like "reddit" or "thanks," etc. The third step is to apply stemming to the words in order to get rid of multiple forms of a word and treat them all as a single word with a common root. For instance, the words "playing" and "played" would be treated as simply "play". The final step is to select a certain number of optimal features and turn the posts or comments into binary feature vectors, in which every entry corresponds to a selected word (feature) and is either 1 or 0, depending on whether the post or comment contains that specific word or not. It should be noted that as the number of selected features is usually fairly large, the feature vectors will naturally be very sparse. Their sparse nature can, therefore, be exploited to speed up the training algorithms. More information on specific functions used for data preprocessing can be found in Appendix A.

3 Proposed Approach

3.1 Feature Extraction

In order for the models to be able to process text data and perform classification on Reddit posts, lexicalized text features were used. The features revolved around the *Bag-of-Words* approach with the complementary option of using either using binary or TF-IDF representation.

The *Bag-of-Words* approach treats documents as a bag of words. The grammar and the order of the words are ignored in this approach. Furthermore, it can be further divided into two sub-approaches, namely the Frequency Method and the Binary Method. In the Frequency method, the multiplicity of each feature in the document is preserved. On the other hand, the Binary method simply keeps information on whether the feature exists within the document or not. This method was used as the baseline approach to evaluate the accuracy of the classifiers.

The TF-IDF (short for term frequency–inverse document frequency) approach is a numerical statistic that conveys the importance of a word to a document in a corpus of documents. TF-IDF was applied to the preprocessed data to extract the TF-IDF scores for each feature.

3.2 Classifier Selection

A brief overview of the motivation behind each classifier as well as their respective evaluation strategies are provided below.

3.2.1 Bernoulli Naïve Bayes

The first classifier that was used and the only one that was fully implemented from scratch. Bernoulli Naïve Bayes is a generative model based on Bayes' theorem [2]. It assumes the conditional independence of the features given a class label. Furthermore, it functions by learning the prior class probabilities, $p(c_j)$, and the conditional probabilities, $p(x_i|c_j)$. The class labels are assigned according to the following:

$$\underset{c}{\operatorname{argmax}}\, p(c) \prod_{i=1}^{m} (p(x_i|c_j))^{x_i} (1 - p(x_i|c_j))^{1-x_i} \tag{1}$$

where x_i indicates the presence or absence of the i^{th} feature in a sample with a given class. For this paper, the multiclass version of Bernoulli Naïve Bayes was implemented since the *1 vs all* approach has a higher time complexity, which is not ideal when paired up with ensemble methods.

3.2.2 Multinomial Naïve Bayes

Another popular method used for text classification due to the high accuracy it yields. This method is very similar to Bernoulli Naïve Bayes as it also selects the class labels which maximize the posterior class probability using (2).

$$\underset{c}{\operatorname{argmax}}\, p(c) \prod_{i=1}^{m} p(x_i|c_j) \tag{2}$$

3.2.3 Linear SVC

An optimized version of Support Vector Machines with a linear kernel. It is used for large data samples with high feature space dimensionality. It features high flexibility in the choice of loss functions, penalties, and multi-classification support. The text data is usually linearly separable [3]. Classifiers like LinearSVC, which attempt to fit multidimensional linear hyper-planes around the data for multi-class separation usually fare quite well in terms of accuracy [3]. Additionally, the text data contains a plethora of features, so transforming them into a higher dimensional space would not necessarily improve LinearSVC's ability to distinguish between labels [3].

3.2.4 Bagging Ensemble with Bernoulli Naïve Bayes

A bootstrap method that uses several individually trained Bernoulli Naïve Bayes classifiers to classify a document and that was implemented from scratch. The data for each classifier is generated by randomly drawing N samples with replacement from the training data set, where N is the size of the original training set. To determine the final classification decision, two voting strategies were experimented with. In the first strategy, the majority voting technique was used. One drawback of this strategy is that it did not perform well when there was a tie between two class labels. To overcome this problem, the second strategy involves obtaining the k-fold cross-validation accuracy of each classifier and providing the scores in the form of weights to the class labels from each classifier.

3.3 Feature Selection

Feature selection was used to enhance the classification process by deleting irrelevant and noisy features. It allows for a subset of all the data to be chosen in order to minimize the complexity of the classification process. In this paper, two feature selection methods were explored.

The first commonly used feature selection method is mutual information. It provides a quantitative score for each feature based on its usefulness in quantifying across the given classes. Using mutual information, the redundant features can be removed from the data based upon the mutual information scores. The second feature selection method is the χ^2-test, which is used in

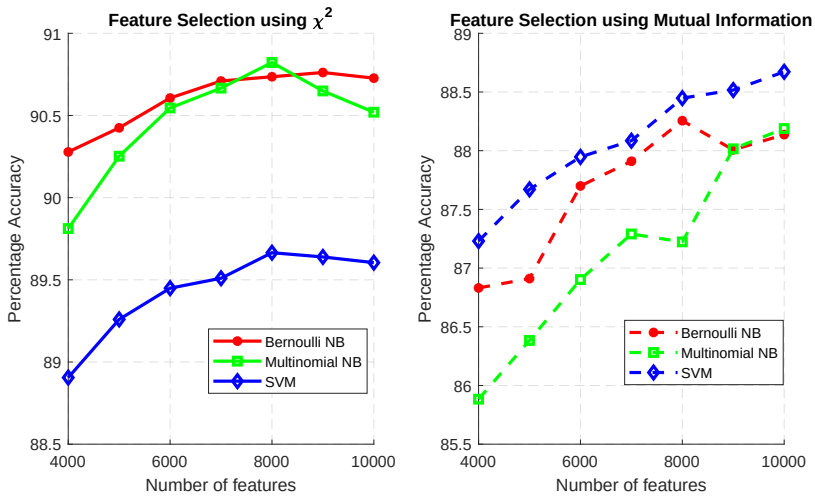

Figure 1: K-Fold Cross-Validation average accuracy as a function of different number of features selected using χ^2-test (left) and mutual information algorithm (right).

statistics to evaluate the independence of two events (variables) [1]. Oftentimes, text data has high dimensionality and not every feature is informative about its target label. The χ^2 statistics are computed for every feature/class label pairing in order to determine the set of features which the target variables are highly dependent upon [1]. The higher the χ^2 value, the more dependence exists between a feature and a target label [1]. K-Cross Validation accuracies were used to decide between mutual information and χ^2-test as well as to find the optimal feature vector size. In figure 1, the χ^2-test is shown to outperform mutual information for all classifiers and attain a peak accuracy at 8000 features.

3.4 Pipeline and Hyper-parameter Tuning

3.4.1 Laplace Smoothing Factor

Laplace smoothing is used to circumvent the problem of having one or more zero probabilities during the training of Naïve Bayes classifiers. In this paper, Laplace smoothing (also known as Lidstone smoothing) was specifically used to smooth the conditional probabilities. The conditional probability, $P(x_i|c_j)$, of finding feature, x_i, given the class, c_j, can be smoothed using the following (3).

$$P(x_i|c_j) = \frac{count(x_i, D_c) + \alpha}{\sum_{x' \in V} count(x') + \alpha * |V|} \tag{3}$$

where α is the Laplace smoothing factor. The smoothing of the conditional probabilities was required for both the Bernoulli NB and the Multinomial NB classifiers. Figure 2 shows the accuracies of these two classifiers as a function of α. The optimal smoothing factors for Multinomial NB and Bernoulli NB were 0.02 and 0.01, respectively.

3.4.2 Pipeline Parameters

Pipeline parameters refer to the input parameters used in **CountVectorizer()**, **TfidfTransformer()**, and **SelectKBest()** from Scikit-Learn [4]. Each parameter change or each combination of parameter changes in either of the three classes resulted in different model accuracies. For instance, assuming all other pipeline parameters remain unchanged, changing the binary flag in CountVectorizer from True to False increased the model accuracy. Thus, to perform efficient pipeline parameter tuning, **GridSearchCV()** was used to exhaustively search the design space[4]. This algorithm's objective was to retrieve the best combination of pipeline parameters that produced the highest model accuracy. The optimal model pipeline parameters used on the test set can be visualized in the appendix section.

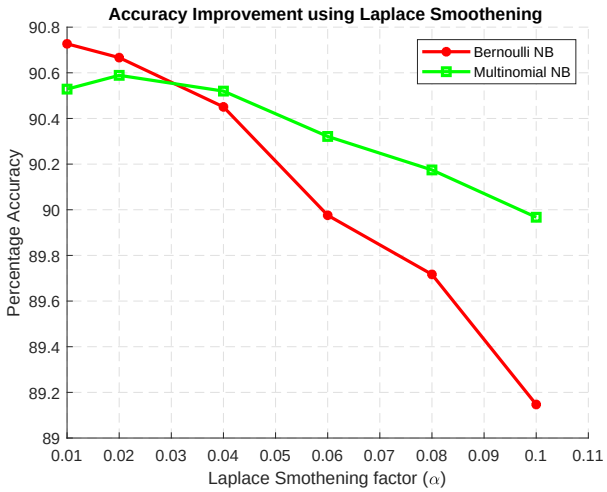

Figure 2: Figure displaying the accuracy of Multinomial and Bernoulli Naïve Bayes as a function of Laplace Smoothing factor.

4 Results

This section describes the results of the various experiments, performed using 8000 features selected using the χ^2 approach, which are summarised below in Table 4. The first and second experiments involved the testing of the Bernoulli NB classifier without feature removal and with the χ^2-test. The χ^2 test feature selection provided the highest K-Fold Cross-Validation average accuracy as well as test set accuracy. As shown in Table 4, the Multinomial NB classifier performs better with TF-IDF combined with the χ^2 test. It also provides higher K-Fold Cross-Validation accuracy than the test set accuracy. Furthermore, using the bagging ensemble method along with 10 Bernoulli classifiers with optimized hyper-parameters yielded the highest K-Fold Cross-Validation accuracy of 92.51%. However, the test set accuracy was found to be significantly lower at a value of 88.5%.

Table 1: Table summarizing the K-Fold Cross-Validation and test set accuracy results

	Experiment	5-Fold Avg. Accuracy	Test Set Accuracy
1.	Bernoulli NB + Binary	89.51%	—
2.	Bernoulli NB + Binary + χ^2-test	90.34%	**91.139**%
3.	Multinomial NB + Frequency	87.65%	—
4.	Multinomial NB + TF-IDF	88.93%	—
5.	Multinomial NB +TF-IDF + χ^2-test	91.13%	88.80%
6.	Linear SVC + TF-IDF + χ^2-test	89.44%	89.01%
7.	Bernoulli Ensemble + TF-IDF+ χ^2-test	92.51%	88.50%

5 Discussion and Conclusion

The results suggest that the the Bernoulli NB classifier with χ^2-test provides the best test set accuracy, which is 91.139%. The Multinomial NB classifier with TF-IDF and χ^2-test feature selection provides a high K-Fold Cross-Validation accuracy but has lower test set accuracy. This suggests that the Multinomial NB classifier exhibits low bias and high variance. Similar behaviour is also observed in the case of the Bernoulli NB Ensemble classifier, for which the K-Fold Cross-Validation accuracy is the highest overall but the test set accuracy is significantly smaller. The test set accuracy of the Ensemble Bagging classifier could be influenced by a number of factors such as the number of classifiers used in the ensemble, the diversity of training data provided to each classifier, and the accuracy of each model in the ensemble [5].
Another noteworthy outcome of this project is that the accuracy of the K-Fold Cross-Validation and that of the test set seemingly adhere to a maximum variability of $\pm 3\%$(which is very small) across all the classifiers as can be seen in Table 4 and Figures 1- 2. This could be due to the fact that some

classifiers may exhibit high variance and low bias and vice versa, in which case the performance of the classifiers would benefit from including multiple types of classifiers in the ensembles.

5.1 Future Work

Several directions for further work based on the results of this project are described below:

- **Improved Ensemble Method**: An ensemble method with a different type of classifiers can be used to tackle the problem with high variance. Also, more voting strategies such as weighted voting, average voting, etc. can be implemented to investigate their impact on model performance.
- **Feature Selection** As can be seen in Figure 1, there is a significant difference in the performance of mutual information and χ^2-test. Therefore, more feature selection methods such as information gain, correlation method, etc. can be implemented to investigate if the accuracy of the given classifiers can be improved.
- **TF-IDF Binarization** TF-IDF feature extraction outperforms the Bag of words model in the case of Multinomial NB. However, it cannot be directly applied to Bernoulli Naïve Bayes as the input to this classifier is required to be in binary format. That being said, a threshold can nonetheless be chosen to scale the TF-IDF scores to a range between 0 and 1.

6 Statement of Contributions

The project tasks were split equally amongst the three team members. Alex Goulet was in charge of data preprocessing and feature selection. Karanvir Sidhu was in charge of the Bernoulli Naive Bayes classifier. Jack Hu was in charge of k-fold cross-validation and the rest of the classifiers from the *scikit-learn* library.

7 Appendix

7.1 Appendix A - Pipeline

Different data preprocessing techniques and classifier pairings generated distinct model accuracies. Thus, in each subsequent section below, in-depth detail is being provided in an additive manner in order to help construct the overall narrative behind the final selected pipeline, feature representation, feature selection method, and classifier pairing.

7.1.1 Data Clean Up

Before calling **CountVectorizer()**, the data preprocessing pipeline also contains a few character by character parsing functions for the purpose of getting rid of unnecessary features. The functions are the following:

- **parseLine()**: Parses each sentence and strips off punctuation, irrelevant words, irrelevant characters, and numbers.
- **isAlphanumeric()**: Dictionary containing relevant characters in a word.
- **isIrrelevant()**: Dictionary containing irrelevant words.
- **isNumber()**: Dictionary containing relevant numbers.

7.1.2 Scikit-Learn Library

After the data had been cleaned up, it is then ready for feature extraction, selection, and evaluation. To perform these tasks, various methods were used from the *scikit-learn* library for machine learning applications[4]. The functions below are listed in their respective calling orders.

- **CountVectorizer.fit_transform()**: CountVectorizer was used to vectorize the training data. It did so by either replacing each sample with a binary vector corresponding to the extracted features or returning a vector of raw frequencies of the features.

- **TfidfTransformer.fit_transform()**: TF-IDF normalizer was used to further modify the count or binary matrix. It returned a term frequency-inverse term frequency representation of the training corpus. The goal was to scale down the impact of frequently occurring features that are empirically less informative than features that occur in a small fraction of the training data.
- **SelectKBest.fit_transform()**: Feature selection with χ^2 Test measured the independence between the features and the class labels. It stripped away features that were most likely independent of the class based on the scored χ^2 value.
- **KFold.split()**: The split() function returned row indices of the training corpus to facilitate slicing of the training data into K desired folds during K-Cross Validation.

References

1. Y. Zhai, W. Song, X. Liu, L. Liu and X. Zhao, "A Chi-Square Statistics Based Feature Selection Method in Text Classification," 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 2018, pp. 160-163, doi: 10.1109/ICSESS.2018.8663882.

2. N. Armanfard, "ECSE 551 - Machine Learning for Engineers Lectures 7, 8, 9 - Naive Bayes," in Machine Learning Lecture Slides.

3. Joachims T. (1998) Text categorization with Support Vector Machines: Learning with many relevant features. In: Nédellec C., Rouveirol C. (eds) Machine Learning: ECML-98. ECML 1998. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), vol 1398. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0026683

4. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay, "Scikit-learn: Machine Learning in Python". 12(85):28252830, 2011.

5. W. Wang, "Some fundamental issues in ensemble methods," 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, 2008, pp. 2243-2250, doi: 10.1109/IJCNN.2008.4634108.