Classification of Reddit Comments using Various Classification Models

Long-Quan Bach 260780847, Peidong Li 260731782, Yimeng Hu 260795862

***Abstract*—For this research, we explored several classifiers to help with subreddit prediction for Reddit comments. The corpus used for the project consists of 70,000 comments with 20 different subreddit categories. We analyzed the performance and accuracy rates of logistic regression, Bernoulli naive Bayes, support vector machine, decision trees and neural network. We also explored different designs for feature selection to improve accuracy, using techniques like chi-square feature selection and mutual information comparisons. It was discovered that in the case of comment classification, naive Bayes and decision trees were less appealing models compared to those of logistic regression and neural networks, with neural networks running BERT language preprocessing reigned supreme as the ideal model for predicting subreddit classification.**

I. Introduction

Text classification problems are widely known in the field of natural language processing. Whether its detecting spam inside emails, categorizing different articles, or understanding sentiment analysis of movie reviews, the amount of research in this field is extensive and broadening every day. This project explores text categorization for the popular online forum Reddit. One of the key features that attract many users to Reddit is its system of subreddit subscription. To organize such a large number of posts, these posts are organized into categories known as "subreddits", providing a smooth user experience to allow them to browse certain types of posts. In each post, users are allowed to create comments to further expand the discussion or address a certain point.

This research explores several different classifiers that could help predict which subreddit a certain comment comes from. This includes Bernoulli naive Bayes, logistic regression, decision trees, support vector machine, and neural networks.

A. Bernoulli naive Bayes

For this research, a fresh implementation of Bernoulli naive Bayes was designed from scratch. naive Bayes is a generative learning model that assumes all its features are independent. It calculates class probabilities $P(c_k)$ and the conditional probability of feature x_i occurring given that the class c_k occurs, $P(x_i|c_k)$. Applying Bayes rule will calculate the probability of a class occurring given a set of input features x.

$$\max_c P(c)\prod_{i=1}^{m} P(x_j|c) \tag{1}$$

The above equation will find the best possible class prediction and calculates $P(c|x)$. Taking the logarithm of this equation will help improve numerical stability.

$$\max_c \log(P(c) + \sum_{i=1}^{m} \log(P(x_j|y)) \tag{2}$$

After this, Laplace smoothing was added to guarantee non-zero probability predictions. In this model, a prior probability of 1/2 was added.

B. Logistic Regression

The logistic regression model was taken from Scikit-Learn's library. Logistic regression is a discriminative classifier that works on calculating the conditional probability directly using gradient descent learning. So for some output y and set of inputs x, we calculate

$$\max_c \frac{1}{1+e^{-y+w^T x}} \tag{3}$$

The weights w from the equation can be adjusted using gradient descent learning based on a loss function.

C. Decision Trees

This project used Scikit-Learn's implementation of decision trees. The algorithm uses top-down recursive induction to fill in leaves. The CART algorithm is used in this model where tree construction is dependent on the amount of information gain at each node (Quinlan).

$$IG(x) = H(D) - H(D|x) \tag{4}$$

The entropy $H(D)$ and conditional entropy $H(D|x)$ are defined as follows:

$$H(D) = \frac{-p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n} \tag{5}$$

$$H(D|x) = \sum_v P(v)H(y|v) \tag{6}$$

A greedy algorithm is run so that the information gain is maximized at each node of this tree and the loss is minimized.

D. Support Vector Machine(SVM)

The support vector machine is a discriminative classifier. Its objective is to find a hyperplane in an N-dimensional space where N is the number of features that distinctly classify the data point(Tang). Scikit-Learn library provides a complete implementation of such a model. Given training data and its corresponding labels $(x_n, y_n), n = 1, .., N, x_n \in R^D, t_n \in \{-1, +1\}$.

SVMs learning consists of the following constrained optimization:

$$\min_{(w,\xi_n)} \frac{1}{2} w^T w + C \sum_{n=1}^{N} \max(1 - w^T x_n t_n, 0)^2 \tag{7}$$

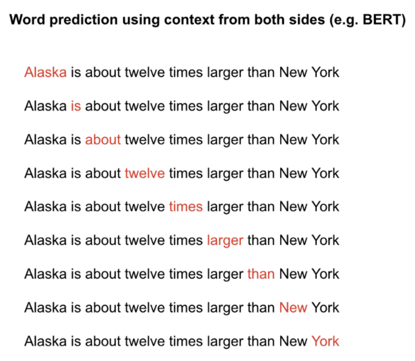
To predict the class label of a test data x:

$$\underset{t}{\operatorname{argmax}} (w^T x) t \tag{8}$$

E. BERT

Otherwise known as **Bidirectional Encoder Representations from Transformers** this model helps pre-training transformers, an intricate neural-network architecture (Devlin). Without going into too many details, the main objectives of BERT circumvent around two ideas:

1) ***Masked Language Model (MLM):*** Given a sentence, MLM selects a sample of tokens and "masks" them, essentially removing them from the sentence and replacing their position by some token variable. It then uses cross-entropy loss to predict what the token variables were before they were replaced (Devlin).



Word prediction using context from both sides (e.g. BERT)

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Alaska is about twelve times larger than New York

Fig. 1. In each of the examples above, the word in red is "masked" and the model will predict the masked word given the words in the sequence.

2) ***Next Sentence Prediction (NSP):*** This next step in the pre-training process involves understanding the relationship between two separate sentences. Given a sentence A, the model is trained to learn what the best sentence should be used after A in its pool of data (Liu et al.).

These two ideas are used to pre-train these transformers. From this, the input is fed through these transformers which results in some output.

II. Related Work

There have been similar work studies related to our project in the past. These works, however, mainly focused on classifying posts to corresponding subreddits as opposed to comments. Stanford undergrads Giel and Kader (2015) used a multinomial naive Bayes and logistic regression as their classifier of choice. They were able to conclude that logistic regression yielded better results in their validation accuracy. Although our work appears to be very similar, their choice of subreddit data was completely different, and this disparity may be enough to cause changes in features and the vocabulary of the models.

III. Dataset

The dataset used in the experiment contains the raw text of 70,000 comments extracted from Reddit. The goal of this project is to develop a supervised classification model that can predict what community a comment came from. Each data point is associated with a unique ID number and a community to which the comment belongs. From Figure 2, it is clear that this is a 20-class classification problem with a nearly balanced dataset. To facilitate future analysis, the labels are encoded into numerical values.

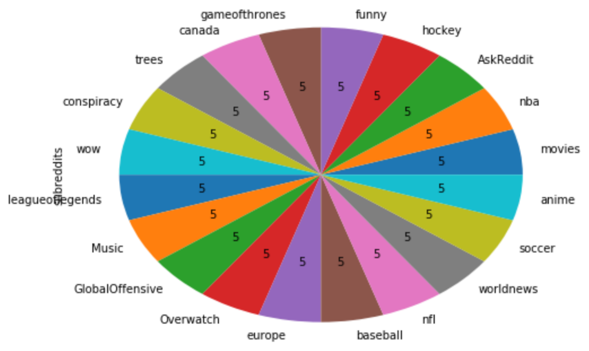


Fig. 2. Label Distributions

A. Cleaning Raw Text

The comment IDs were first removed from the dataset since this feature was randomly assigned to each data point and is therefore irrelevant in categorizing the comments. Next, we cleaned the comment by removing HTML links, non-word characters, special characters and stop words. Then turned every word into lowercase before tokenization. There was a noticeable decrease in the accuracy after lemmatization, thus only stemming was applied to the text comment. Moreover, we found that limiting the word within a certain range of frequency helped to remove noise and improves accuracy. Therefore the features were limited to a maximum of 50,000 most frequent words.

B. Vectorization

Various word embedding methods were provided in Scikit-Learn's library. The Bag-of-Words and TF-IDF text extractors were chosen for text vectorization.

1) Bag-of-Words: BOW is an algorithm that simply converts the input text data to a matrix of token counts. Before they are fed to the classifier, each vector of the token counts is normalized such that all elements of the vector add up to one. Thus, the frequency of each word is effectively converted to represent the probabilities of those words' occurrence in the document.

2) TF-IDF: Short for Term Frequency Inverse Document Frequency, it assigns each word a weight, TF-IDF, accross the whole dataset calculated as following:

$$tf(t,d) = \frac{f_d(t)}{\max_{w \in d} f_d(w)} \tag{9}$$

$$idf(t,d) = \ln \frac{| D |}{| \{d \in D : t \in d\} |} \tag{10}$$

$$tf - idf(t,d,D) = tf(t,d) * idf(t,d) \tag{11}$$

$f_d(t)$:= frequency of term t in document d
D := corpus of document

C. Feature extraction

In order to extract a more informative vocabulary set, we further proceeded with the following three feature extraction techniques separately.

1) N-Gram: Depending on the context, the meaning of a word can change. This can highly impact the classification problem as it changes how the classifier perceives certain features. A sequence of words can be more helpful as a feature opposed to individual words. To recognize the difference, the vocabulary can be expanded to multiple tokens that are grouped together as a singular feature.

2) Mutual Information(MI): MI refers to the measure of the mutual dependence between two random variables. Given a variable X_j and a target Y

$$I(j) = \int_{X_j} \int_{Y} p(x_j,y) \ln \frac{p(x_j,y)}{p(x_j)p(y)} dxdy \tag{12}$$

In our case, the mutual dependency was calculated between the target value and each word in the vocabulary. Words that contribute little to the classification problem produced low mutual information scores and are theoretically independent with the target value, therefore these words can be removed from the set of features.

3) Chi Square Statistic: Like mutual information feature extraction, the chi-square test measures the dependency between two random variables. In our case, the chi-square statistic is measured between the target values and the features. Features were eliminated based on their likelihood of being independent of target values and therefore irrelevant for classification.

IV. Result

A multitude of tests were performed on the models which include (1) the performance comparison of different vectorization methods and feature extraction techniques applied on various classifiers, (2) the performance comparison between different classifiers, (3) classifier selection and attempt on accuracy improvement with various parameters, and (4) the use of neural networks to generate better models. Throughout this project we used the k-fold cross-validation method to train and measure performance of our models and to demonstrate our results.

A. Vectorization Methods

During the pre-processing of raw training data, we have attempted to use the two vectorization methods mentioned above. Performance comparison is shown in the table below:

	BOW	TF-IDF
Logistic Regression	52.471	54.200
Bernoulli Naive Bayes	51.214	53.493
Decision Tree	30.364	32.56

Table 1: The Percentage of Performance Accuracy of Classifiers against BOW and TF-IDF vectorization methods

As you can see TF-IDF generates better performance compared to BOW across all classifiers we have used in the sklearn library, and hence we decided to carry on our training process with this approach.

B. Classifier Selection

After pre-processing with TF-IDF and removal of stopwords, we fitted our data with the stipulated classifiers and tested the prediction accuracy with k-fold cross-validation. Note that we have excluded SVM from this process due to high running time (about 1 hour per fold) and low model accuracy. In this case, we've tested the accuracy outcome with 5, 10, 15, 20 fold cross-validation, with 50000 features and n-gram=3. The results are listed below:

	5-fold	10-fold	15-fold	20-fold
Logistic Regression	54.986	54.161	54.922	54.636
Bernoulli Naive Bayes	53.114	52.481	51.683	51.435
Decision Tree	30.397	30.097	30.027	29.974

Table 2: The Percentage of performance Accuracy of Classifiers against 5, 10, 15 and 20-fold Cross-Validation

We can conclude that logistic regression and naive Bayes perform better in general compared to other classifiers, and this would be the model we carry on for optimization with parameter selection.

C. Feature Extraction Techniques

Since we know logistic regression yields better performance overall, we analyze different possible feature extraction techniques with this model to find optimal values for different parameters.

1) Mutual Information Classification: From the vocabulary set of 50,000 words, 40,000 with the highest dependency were selected. The logistic regression model with the reduced vocabulary resulted in a 53.228% accuracy score. We can conclude the the accuracy rate drops when using mutual information to reduce our vocabulary.

2) Chi Square Statistic: The chi-square statistic between the target values and each feature is measured. Figure 3 shows that out of 50,000 words, the accuracy is the highest with 12,500 selected best features.

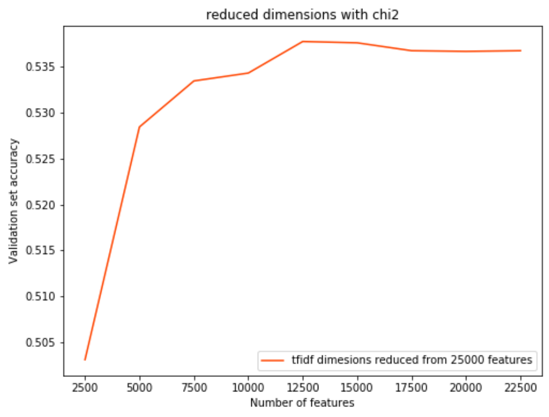


Fig. 3. Chi-Square

It is interesting to notice that in general the accuracy decreases after applying chi-square statistic analysis.Hence we decided not to go on with this approach.

3) N-Gram: We tested logistic regression with unigram, bigram and trigram. By figure 4, it is clear that we obtained the highest accuracy with unigram vocabulary.

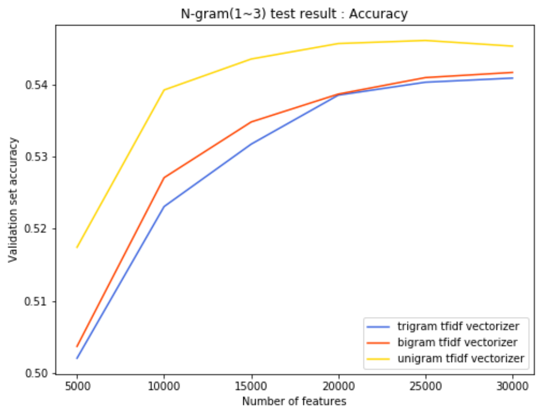


Fig. 4. Difference in performance by 1,2,3-gram

D. One Step Further: BERT Pre-processing

It has come to our realization that there is a limit in prediction accuracy with our existing models. To find a better model, we have to take one step further and research on state-of-the-art natural language processing techniques and libraries. After researching online, we came across a library called ktrain. It provides more sophisticated data processing and modelling techniques that we could utilize (refer to section 1.D). Using BERT as the pre-processing mode and some different parameters, we obtained much more promising models compared to those we have implemented.

	Epochs	Val %	Max Features	Accuracy
Trial 1	1	0.2	50000	0.5930
Trial 2	1	0.1	40000	0.5881
Trial 3	1	0.1	50000	0.5961
Trial 4	1	0.1	60000	0.5974
Trial 5	1	0.1	50000	0.5961
Trial 6	3	0.05	62000	0.6217

Table 3: Performance Accuracy of neural networks running BERT language pre-processing with various parameters

By utilizing cloud servers and GPU on Kaggle kernel, we could complete one cycle in each training process in about 1.5 hours instead of 48 hours on an average laptop. It is not practical to find the trend of performance on a large set of parameters due to time constraints, we made an informed guess about possible parameters and our first trial generated a better model compared to all our previous approach. We then made a few more attempts with different parameters while observing the trend, and eventually landed on the model we've specified in our code.

V. Discussion

A. Ethnic Concern

The datasets we used are from the Reddit website, which gathered data from comments contributed by registered users around the world. There might be inappropriate or even abusive words from people of different level of education and different cultural backgrounds. This dataset may be used for various academic research or industrial experiment. One might find the content offensive to certain groups or the general public.

B. Possible Improvement

1) The hyperparameters of the two classifiers from sci-kit learn and neural networks were all set by default. It is possible to run a grid search to find the best hyperparameters.
2) The default stopwords list could be non-exhaustive, we could investigate more on words that does not have any values during categorization to further reduce noise.
3) There could be other possible features related to comment classification such as the habit in punctuation, sentence structures and length that were not assessed in our analysis.
4) We were not able to iterate through all possible parameters for neural nets due to hardware and time constraints. Given better hardware and more time we could obtain the optimum parameters for this particular dataset to achieve maximum accuracy. Also, the accuracy of data is increased with the sample size of the training set, so a larger training would theoretically yield a better result, which is more relevant to real-life scenarios. As pre-processing methods such as BERT and libraries such as ktrain improve over time or if a new neural nets approach appears, we could also see a boost in prediction accuracy.

VI. Conclusion

In this project, we explored five different classification models, Bernoulli naive Bayes, Logistic Regression, Decision Trees, Support Vector Machine(SVM) and neural networks running BERT language pre-processing. We compared their performance in terms of predictive accuracy. Empirically, we illustrated that the prediction accuracy is heavily influenced by model and feature selection. The trend of change in accuracy is homogeneous across different models. Additionally, we highlighted that the practices of data pre-processing using different vectorization methods, parameter selection using feature extraction techniques can significantly increase the predictive accuracy of machine learning algorithms. Finally, we ventured beyond the project requirement and investigated more leading natural language processing techniques for better models beyond our course coverage. This project revolves more around exploring performance on different models with parameter selection as the variable other than a detailed understanding of specific features and their relationship with each other because there is a large number of features and there is large uncertainty for the appearance of words in a language. For some models with fast running time such as Bernoulli naive Bayes, we could find the optimal prediction parameters under a reasonable period and obtain a model close to the full potential of this model with the specific data set. For models with long running time such as neural networks running BERT language pre-processing, we could no longer achieve an exhaustive plot but make informed guesses based on our experiences and analysis of data.

VII. Statement of Contribution

All three members contributed to the project equally. The main framework of Bernoulli naive Bayes was implemented by Long-Quan Bach. The Sci-Kit Learn classifiers were implemented by Yimeng Hu and the testing pipelines and neural networks running BERT language pre-processing were implemented by Peidong Li. We all contributed to the execution of experiments and writing of this report.

References

[1] Anderson, K. E. (2015). Ask me anything: what is Reddit?. Library Hi Tech News, 32(5), 8-11.

[2] Ng, A.Y. & Jordan, M.I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in neural information processing systems*. (pp. 841-848).

[3] Quinlan, J. R. (2014). C4. 5: programs for machine learning. Elsevier.

[4] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[5] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

[6] Giel, A., NeCamp, J., & Kader, H. (2015). CS 229: r/Classifier-Subreddit Text Classification.

[7] Tang, Yichuan Deep Learning using Linear Support Vector Machinse. arXiv:1306.0239v4