Introduction to Machine Learning
Mini Project 2

Gian Favero[1] **Hieu Thien Hoang**[2] **Maxime Favreau-Vachon**[1]
[1] McGill University [2]École de Technologie Supérieure
gian.favero@mila.quebec
thien.hoang@mail.mcgill.ca
maxime.favreau-vachon@mail.mcgill.ca

Abstract

This paper presents a comprehensive exploration of text classification in the context of Reddit posts originating from four subreddit groups: Toronto, Paris, Montreal, and London. The class of posts from these groups is attempted to be learned using various classification models, including a Naive Bayes model constructed from scratch, and models made available via scikit-learn such as Support Vector Machines (SVM), Logistic Regression, and Random Forest. These models are coupled with a meticulous preprocessing pipeline involving, stop-word removal, lemmatization, and information gain. A detailed analysis of model performance is conducted through a 5-fold cross-validation process, revealing notable improvements with preprocessing steps such as common word removal, mutual information filtering, and word replacement. Each model demonstrates similar challenges in accurately classifying posts from the classes with similarities in linguistic and geopolitical context (e.g., Montreal versus Toronto or Paris). This challenge was best overcome with a model stacking approach that combined SVM, Logistic Regression, and Random Forest classifiers with a majority vote. This ensemble approach, combined with strict preprocessing, yielded a substantial increase in accuracy. The final model stack achieves an accuracy of 72% on the Kaggle evaluation set.

1 Introduction

Text classification is a crucial aspect of machine learning applications, it empowers systems to automatically categorize and make sense of vast amounts of textual data, providing valuable insights for decision-making and enhancing user experiences. The dataset of this project comprises posts and comments from Reddit, originating from four subreddits (London, Montreal, Paris, and Toronto), and are in either English or French. Contents from each subreddit contribute approximately 25% of the data. The major goal of this project is to enhance our experience working with text data and machine learning using scikit-learn. This involves developing our preprocessing strategies, constructing Bernoulli Naïve Bayes (BNB) from scratch, comparing the outcomes with other built-in Scikit-learn packages for classification, implementing model validation, and exploring ensemble models. In preprocessing, we propose different approaches such as common word removal, mutual information filtering, and word replacement. Our findings reveal that word replacement can enhance the results. In the 5-fold cross-validation process, we employ BNB, Logistic Regression, Random Forest, and ensemble models. The outcomes highlight the superior performance of the stacked model, which delivers the best results. In this report, we investigate the performance of our built-from-scratch BNB and our best performance model, i.e., the stacked model in the Results part.

2 Datasets

The qualitative data dataset comprises textual samples collected from posts and comments on four different subreddits. Each sample represents user-generated content and discussions from these online communities and include a variety of different types of characters and textual elements such as capital letters, lowercase letters, punctuation marks, and special characters. The dataset is used for a multi-class classification task where the labels correspond to the city of origin for each subreddit, which includes Toronto, Paris, Montreal, and London.

2.1 Class distribution

Figure 1 depicts the distribution of classes within each dataset, revealing a well-balanced distribution with approximately 25% of posts and comments falling into each class.

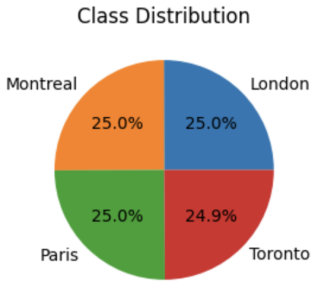

Figure 1: Distribution of the four subreddits

2.2 Preprocessing

For data preprocessing before classification, we employed several essential steps to improve the accuracy of the classification models. First, we converted all text to lowercase for uniformity. Next, we removed all punctuation marks and special characters to ensure a clean and consistent text. Additionally, stop words in both English and French were eliminated to focus on more meaningful content. To further refine the text, we employed lemmatization for verbs and nouns, which helped standardize word forms. Then, we employed a word replacement strategy in which a dictionary of domain specific words for each subreddit (e.g., “CN Tower”, “Quebec”, “mtl”) were directly replaced by the class label itself. The most commonly shared words across different classes were identified and were subsequently excluded. By removing these shared words, the distinct characteristics of each class are brought into sharper focus and contributes to a more effective classification process, as the remaining unique and class-specific terms become more influential in distinguishing one class from another. Finally, an information gain approach was used to determine the most relevant subset of words necessary for the subreddit classification. The mutual information of every word in the dataset was calculated and filtered such that only the words with the highest information gain were considered. Based on experimental trials, it was found that the most informative balance was achieved removing 300 of the most commonly used words, and keeping the top 3500 words based on information gain. These preprocessing techniques collectively contributed to the improvement of our dataset’s quality and prepared it for more effective classification in the downstream classification task.

3 Proposed Approach

In the scope of the project, we employed a variety of classifier models to compare their performance with our dataset. This analysis included the development of a Bernoulli Naive Bayes model from scratch but also the utilization of established machine learning models provided by the scikit-learn library, such as logistic regression, random forest, multinomial Naive Bayes, and support vector classifiers [1].

Before applying these models, we preprocessed the data as detailed in the previous section. Supplementary to this, we selected between two text vectorization techniques namely TfidfVectorizer and CountVectorizer. These transformers serve the purpose of converting

text data into numerical feature vectors. The main difference between them lies in their approach to term representation. TfidfVectorizer considers both term frequency and inverse document frequency, emphasizing the importance of terms across the entire dataset, while CountVectorizer quantifies term frequency without weighing their significance in the dataset [2].

To ensure a robust evaluation, we utilized a StratifiedKFold cross-validation, a technique that divides the dataset into subsets while maintaining class distribution, thereby mitigating bias in the assessment process [3]. Additionally, we used Pipeline from scikit-Learn, a tool that combines data transformers and classifiers into a unified workflow, automating data preprocessing and ensuring consistent application of these steps before classification [4]. Furthermore, we employed GridSearchCV to systematically optimize hyperparameters, fine-tuning the models' configurations and enhancing their predictive performance [5]. This comprehensive approach facilitated a detailed comparison of the models and their effectiveness in classifying the dataset.

3.1 Classification Models

Logistic Regression stands out as a widely adopted and adaptable algorithm suitable for text classification tasks. It operates as a linear model, making predictions about the likelihood of a text's association with a particular class using a logistic function. It can be used in both binary and multi-class classification scenarios, and it can incorporate regularization methods to guard against overfitting. This algorithm is effective in capturing linear relationships between the words in the text and their respective classes. However, it may have limitations in capturing intricate nonlinear patterns within the text [6].

Support Vector Machines (SVMs) are adaptable algorithms applicable to text classification. Their principle revolves around identifying the optimal hyperplane that maximizes the separation margin between data points of distinct classes. SVMs excel in managing both linear and nonlinear classification challenges and offer the flexibility to employ diverse kernels for data transformation into higher-dimensional spaces. The challenge with text classification often requires managing an abundance of features, frequently numbering above 10,000. In this context, SVMs offer a valuable advantage. These algorithms employ overfitting protection mechanisms that aren't inherently tied to the number of features. This distinctive trait positions SVMs as well-suited for managing the challenges posed by extensive feature spaces [7].

Random Forests operate on the principle of partitioning the data into more compact and internally consistent subsets, guided by specific criteria. Class labels are assigned to the terminal leaf nodes of these trees. They are an ensemble learning method that combines multiple decision trees to improve classification performance. They are robust to overfitting, can handle high-dimensional data, and are less sensitive to outliers. An added advantage is the reduced variance compared to standard decision trees, which contributes to the model reliability [6].

AdaBoost operates as an iterative ensemble technique. It constructs a robust classifier by amalgamating several weak classifiers, resulting in a highly accurate strong classifier. The fundamental idea of AdaBoost involves adjusting the weights of classifiers and training data samples in each iteration, thereby guaranteeing precise predictions for atypical observations. Any machine learning algorithm capable of using weights on the training set can serve as a base classifier in AdaBoost. [8]

Finally, we constructed a Bernoulli Naive Bayes classifier from the ground up, while also employing a Multinomial Naive Bayes classifier from the scikit-learn library for comparative analysis. In the Bernoulli model, a text corpus is viewed as originating from a multidimensional Bernoulli distribution, similar to a weighted coin flip where words are either present or absent. In contrast, the multinomial model represents a text corpus as a result of drawing words from a multinomial distribution, similar to rolling a weighted dice with probabilities tied to word frequencies. This leads to differences in text representation: Bernoulli employs a binary vector to denote word presence, while Multinomial uses an integer vector to convey word frequency. Furthermore, Multinomial considers the multiple occurrences of words, making it more suitable for longer texts, while Bernoulli is apt for shorter ones. Additionally, the models behave differently with common words like 'the', where Bernoulli assigns high

probabilities, and multinomial calculates probabilities based on relative frequencies, resulting in lower likelihood for such words in a given class [9]. Hence, the importance of removing common words before using a Bernouilli Naive Bayes Classifier.

4 Results

A pipeline was constructed to evaluate the models via a 5-fold cross validation process using a stratified k-fold object. Five independent models were considered along with a sixth method which utilized model stacking of SVM, logistic regression, and random forest classifiers with a majority vote. Various ablations of the preprocessing pipeline were studied to determine the most optimal combination of dictionary word replacement, common word removal, and mutual information filtering.

	SVM	**LR**	**RFC**	**NB**	**BNB**	**Boost**	**Stack**
No Preprocessing	63.25	61.62	66.49	62.17	65.09	59.25	59.95
Basic Preprocessing	63.28	61.62	65.23	63.56	65.37	66.48	63.98
Common Word Removal	69.82	69.12	68.85	67.46	65.09	63.70	69.26
Mutual Information	68.43	68.15	67.46	66.90	64.53	64.53	69.40
Word Replacement	71.21	70.79	68.85	69.40	65.51	65.37	71.07
Kaggle	-	-	-	-	-	-	**72.29**

Table 1: Ablations and accuracies

Table 1 shows 5-fold cross validation accuracies for a series of models fitted via a grid search. Initially, the models were simply trained and tuned on the raw data. Stage 1 of preprocessing involved converting all text to lowercase, accent, symbol, stop-word and punctuation removal, and lemmatization. While minor accuracy increases were seen here, a drastic uptick was seen consequent to the removal of the 300 most commonly occurring words in the dataset (aggregate, across all classes). From there, a third stage of preprocessing was done to select the top 3500 words in the dataset based on information gain. Finally, a list of words was procured based on domain knowledge and a word replacement strategy was executed to replace these buzzwords with the word of the subreddit itself (e.g., "McGill" to "montreal").

SVM, logistic regression, and random forest classification models were chosen in the model stack in an attempt to minimize subclass errors as much as possible. Each of the models brought forth strength in predicting the Toronto, London, and Paris subreddits (approximately 75% accurate in each), but struggled with Montreal ($< 50\%$ accurate). In observing the confusion matrices aggregated over the cross validation procedure, it was hypothesized that stacking the models could aid in offsetting the misclassification of Montreal Reddit posts, as each of SVM, logistic regression, and random forest classifiers seemed to have unique struggles. This hypothesis was found to be true as the model stack with majority voting generalized most optimally on the unseen Kaggle test set with an accuracy of approximately 72%. While the SVM model performed best according to the cross validation accuracy, it could not solely outperform the model stack, hence showing a strength in choosing a more general model for unseen data.

Figure 2 shows the aggregate confusion matrix over the 5-fold cross validation of the Naive Bayes model that was made from scratch. For reference, the class labels are Toronto, London, Paris, and Montreal, respectively. Clearly, the classifier struggles the most when predicting posts that originate from the Montreal subreddit. Sharing a language with all other classes (English and French) represents a major barrier to accurate classification of posts from Montreal, and as such, a majority of the preprocessing effort went towards making the Montreal class as distinguished as possible.

Figure 3 shows the aggregate confusion matrix over the 5-fold cross validation of the stacked model. While the information gain and word replacement approaches produced marginal improvements, a massive increase in performance is indicated in Table 1 when common words across all four classes are removed. A clear advantage in accuracy of posts originating from the Montreal subreddit is seen here.

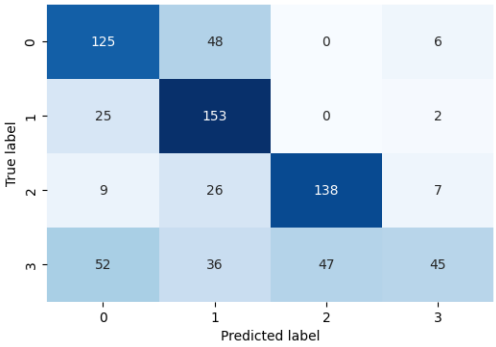

Figure 2: Bernoulli Naive Bayes confusion matrix during 5-fold cross validation

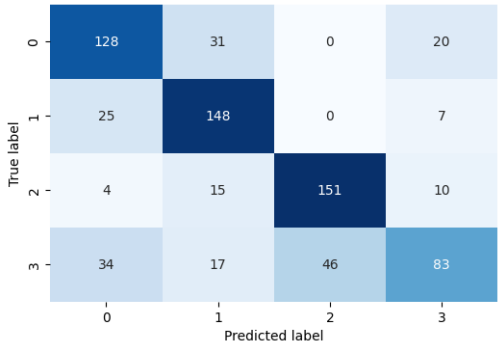

Figure 3: Ensemble confusion matrix during 5-fold cross validation

While automated hyperparameter tuning via a grid search (e.g., regularization, iterations, CountVectorizer lengths) produced marginally increased accuracy, it was observed in this application that proper preprocessing of the text was of most importance in optimizing classification.

5 Discussion and Conclusion

The results of the text classification project revealed notable challenges in accurately classifying posts from the Montreal subreddit. Despite employing various preprocessing techniques and utilizing different classification models, the Montreal class consistently exhibited lower accuracy compared to other classes. One contributing factor could be the linguistic and geopolitical similarities between Montreal and Toronto. Even though Montreal has French as its official language, there is a significant proportion of English speakers and both Montreal and Toronto are in Canada which could explain the blend of linguistic influences that contribute to overlapping content and contribute to the complexities of distinguishing linguistic patterns. Similarly, distinguishing between Montreal and Paris poses a challenge, given that both cities are predominantly French-speaking, and Montreal exhibits a strong French influence. To address this, significant efforts were made in preprocessing, including common word removal, mutual information filtering, and word replacement, with a focus on making the Montreal class more distinct. However, the challenges persisted.

In the case of the ensemble algorithm, which combined SVM, logistic regression, and random forest classifiers, it demonstrated substantial improvement, particularly in classifying Montreal posts. This improvement suggests that the combination of diverse classifiers, each capturing different aspects of the data, helped mitigate the misclassification issue. For the Bernoulli Naive Bayes model constructed from scratch, the challenges in accurately predicting Montreal posts persisted, indicating potential limitations in capturing the complex linguistic patterns unique to Montreal. You can see in the appendix that, even though algorithms using Adaboost did not outperform the accuracy of the ensemble, they are the best working algorithms to correctly classify the Montreal class. This can be easily explain as one of the key advantages of Adaboost is its ability to focus on misclassified instances and assign higher weights to them during subsequent iterations, effectively improving the model's performance on challenging data points.

Additionally, for each algorithm, adjustments might include fine-tuning hyperparameters through more comprehensive grid searches, exploring advanced natural language processing techniques, or incorporating features specific to the linguistic nuances of Montreal.

6 Statement of contributions

Every team member contributed equally to the construction of the experiments run and to this report.

References

[1] Susan Li, "Multi-class text classification model comparison and selection," https://towardsdatascience.com/multi-class-text-classification-model-comparison-and-selection-5eb066197568#:~:text=Logistic%20regression%20is%20a%20simple,easily%20generalized%20to%20multiple%20classes.&text=We%20achieve%20an%20accuracy%20score,and%201%25%20lower%20than%20SVM., 2018.

[2] D. Kaplan, "Machine learning 101: Countvectorizer vs tfidfvectorizer," https://enjoymachinelearning.com/blog/countvectorizer-vs-tfidfvectorizer/, 2022.

[3] Scikit-Learn Developpers, "sklearn model selection stratifiedkfold scikit-learn 0.21.3 documentation," https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html, 2019.

[4] ——, "sklearn pipeline pipeline scikit-learn 0.24.1 documentation," https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html, 2019.

[5] ——, "sklearn model selection gridsearchcv scikit-learn 0.22 documentation," https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, 2019.

[6] Kamran Kowsari, "Text classification algorithms: A survey," https://medium.com/text-classification-algorithms/text-classification-algorithms-a-survey-a215b7ab7e2d, 2019.

[7] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European Conference on Machine Learning (ECML)*. Berlin: Springer, 1998, pp. 137–142.

[8] Avinash Navlani, "Adaboost classifier algorithms using python sklearn tutorial," https://www.datacamp.com/tutorial/adaboost-classifier-python, 2018.

[9] H. Shimodaira, "Chapter 7 text classification using naive bayes." University of Edinburgh School of informatics, 2020.