COMP 551: Mini-Project 2: Machine Learning

Luis Medrano del Rosal (260899458)
Manoj Krishna Venkatesan (260899964)
Israel Hernandez Quinto (260854344)
Group 7
McGill University
March 12, 2020

***Abstract*— Text classification algorithms are the basis for Natural Language Processing, which is being extensively used throughout many sectors of the industry. Thus, understanding which Machine Learning algorithms and optimization techniques to implement into text classification become indispensable for proper labeling. In this report, several machine learning algorithms and combined models were implemented into two datasets: 20-newsgroup [1] and IMDB sentiment [2]. To each of these datasets the following Scikit-Learn [3] machine learning models were implemented: Logistic Regression, Decision Trees, Support Vector Machine (SVM), Adaboost (using Decision Trees as weak learner), Random Forests, and Naive Bayes. Apart from these models, four alternative algorithms were proposed as experiments to review performance variation. The first experiment consisted of using Adaboost with different weak learners. The second experiment used five classifier models and through a voting process the final prediction was obtained. The third experiment, which was implemented only to the 20-newsgroup dataset, was based on hierarchical classification. Finally, the fourth experiment, which was implemented only towards the IMDB dataset, consisted on implementing an Neural Network. The results showed that for the 20-newsgroup the best accuracy came from implementing the combined vote classification mode. Nevertheless, a close second-best accuracy was achieved using Naive Bayes. Meanwhile, for the IMDB dataset, the best accuracy came from implementing the SVM model.**

Keywords: Text Classification, Logistic Regression, Decision Trees, SVM, Random Forests, Naive Bayes, 20-newsgroup, IMDB.

[2]The authors are with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC H3A 2K6, CANADA Project Repository at $github.com/LuisMedrano1/Miniproject2-ML$

I. INTRODUCTION

The objective of this project is to implement different classification models to text classification datasets. Two different datasets were used to experiment the different models: 20-newsgroup and IMDB datasets. The first dataset consists of 20 target classes and the second one consists of only 2. These datasets are further described in the Dataset and Setup section. The following models were applied: Logistic Regression, Decision Trees, Support Vector Machine (SVM), Adaboost (using Decision Trees as weak learner), Random Forests, and Naive Bayes. Also, four extra experiments were implemented into an attempt to obtain better insight: Adaboost with different weak learners, combined voting classifier, hierarchical classification, and Neural-Network. Each of the datasets were preprocessed through vectorization and the models were hyper tuned. The results showed that the best results for 20-newsgroup came from Naive Bayes (70.10%) and combined voting model (70.3797%); while, for IMDB the higher accuracy was obtained from Neural-Network (90.80%) and SVM (also 90.8280%).

II. RELATED WORK

Our Mini-Project 1 focused on binary classification problems; however, real life problems don't necessarily stay within this realm; as an article of *towards data science* [4] suggests, there are many commercial applications that involve multi-class text classifications. As Bansal [5] explains, text classification can be found spread everywhere in the forms of methods for: Understanding audience sentiment on social media, auto tagging costumer queries, categorizing news articles into topics, etc. In order to better cope with the widely-spread problem of multi-class text classification, a huge amount of research has been done around it. The following two examples show developments

that demonstrate the progress that has been done by investigating multi-class text classification.

Hassan H. et al [6] worked on simple but efficient algorithm for multi-class text classification called Feature Weighting Classifier (FWC). This method used information gain to estimate feature weights for each class in a classifier that needs only one pass over the dataset for computing the feature frequencies of each class, and thus has a linear memory usage. The performance of this method can be equal or even better than Naive Bayes and linear SVM. In their journal, Hassan H. et al show a case in which FWC obtained a similar performance than a state of the art SVM implementation, while taking 13 seconds to train against 15 minutes of training for the SVM.

Mengdi et al. [7] decided to work in the field of social media, particularly twitter, to demonstrate the advantage of multi-class over binary classification. In this exercise, they worked on what they called Twitter Sentiment Analysis by studying the emojis of a dataset composed by tweets produced as reaction of the 2016 Orlando nightclub shooting. Rather than classifying in negative and positive reactions, they split the reactions into highly positive, positive, neutral, negative and highly negative.After implementing the models and performing tests, the team found performances with over 60% of accuracy, which is quite acceptable even when compared with other binary classification methodologies. However, the main advantage of their implementation, is that their predictions provide with more information about the sentiment of the public towards different topics that are discussed in Twitter. This could allow governments or other agencies to understand the distribution of these sentiments in a more realistic way and with a higher precision.

III. DATASET AND SETUP

For both datasets, all the instances were vectorized in a sparse matrix (to reduce computational cost) and normalized using the term-frequency and inverse document-frequency procedures with the TfidfVectorizer function from sklearn. Also, for the 20-newsgroup dataset, the header, footer and quotes were not considered. Finally, the "stop words" were eliminated to reduce noise in the both datasets.

A. 20 Newsgroup Dataset Distribution

After preprocessing, the 20 newsgroups training dataset had a shape of 11,314 instances (40% of total instances) and 101,322 features. The target distribution

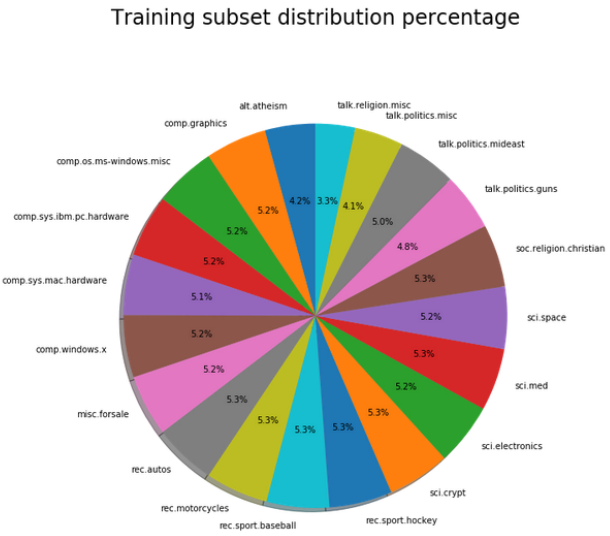

Fig. 1. 20 Newsgroup Training Dataset Distribution Percentage

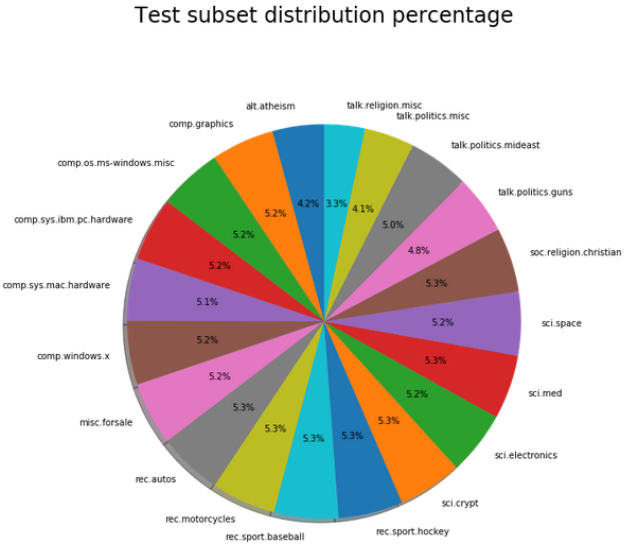

Fig. 2. 20 Newsgroup Test Dataset Distribution Percentage

can be seen in Fig. 1 below. As seen in Fig. 1, there were a total of 20 target classes which were almost distributed equally, ranging from 3.3% to 5.3% of the instances. The same preprocessing was performed for the test dataset, which consisted of 7,532 instances (60% of total instances). The distribution for the test data set can be seen in Fig. 2 below, notice that it has a similar distribution as the train dataset.

B. IMDB Dataset Distribution

Meanwhile, the IMDB dataset has only two target classes, i.e. it's a binary classification dataset. In total there were 25,000 training instances (50% of total instances) and 25,000 test instances (50% of total

instances). In total there were 75,000 features and, as seen in Fig. 3, the dataset target distribution was evenly distributed..

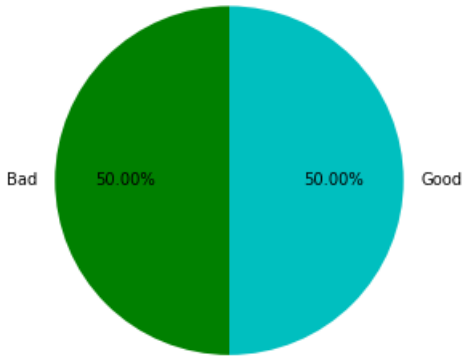

Fig. 3. 20 IMDB Dataset Distribution Percentage

IV. PROPOSED APPROACH

For both datasets the following models were used to predict the targets: Logistic Regression, Decision Tree, SVM, Adaboost (with Decision Tree as weak learner), Random Forest, and Multinomial Naive Bayes. All these models were directly taken from sklearn library. A validation pipeline, along with the Grid Search Cross Validation function from sklearn, was constructed to tune the hyperparameters of each of the models. The following parameters were tuned per each model:

1) Logistic Regression and SVM:
   a) Inverse Regularization parameter
2) Decision Trees:
   a) Criterion: Gini or Entropy
   b) Max tree depth
3) Adaboost:
   a) Learning rate
   b) Number of estimators
4) Random Forest:
   a) Max depth
   b) Number of estimators

After hyperparameter tuning, for the 20-newsgroup, the models were trained with the training subset and tested with the test subset. Meanwhile, the performance for the IMDB dataset was obtained from a 5-fold cross validation. The performance of each of these models is presented in the Results section of this report. Besides these basic models, three experiments were performed. The first experiment consisted of using Adaboost with the following weak learners: SVM and Multinomial Naive Bayes. This experiment was implemented to see the effects of having different base learners for Adaboost. According to the work Ting and Zheng [8] there is no significant increase while using Adaboost with Multinomial NB. Also, Li, et al. [9] comment that applying Adaboost to SVM tend to make the model more inefficient due to the errors being highly correlated. Thus, it would be expected to obtain a lower accuracy than the original models SVM and Multinomial NB. The second experiment consisted of using the prediction of multiple classifiers, and through voting a final prediction would be given. The five classifiers used were: Logistic Regression, SVM, Random Forests, Adaboost using SVM as weak learner, and Multinomial Naive Bayes. In case there exists a tie, the final decision was made by SVM's prediction. Saha and Ekbal [10] mention that using an ensemble method, such as this one, results in an increased accuracy compared to the individual classifiers. The combined classifiers method proposed can be seen in a graphical interpretation in Fig. 4 below.

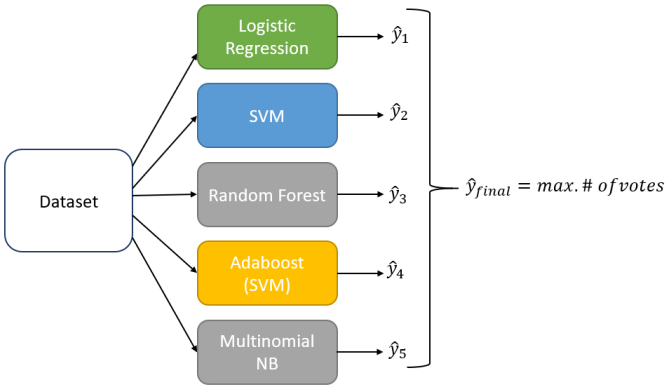

Fig. 4. Combined Classifiers Prediction

The third experiment consisted of performing a Hierarchical Classification, but it was restricted only to the 20 Newsgroups Dataset. This approach was inspired in the work of Ying and Run-Ying [11], where they construct a hierarchy classification tree by categorizing and organizing instances into groups. These groupings are structured into hierarchy levels and in each level a prediction is performed with a base learner. The experiment consists of using two hierarchical trees and two base learners. The two proposed hierarchical trees are found below in Fig. 5 and Fig. 6. Notice that the first proposed tree is based on the work of Graovac, et. al [12], where they used an "n-gram" algorithm to

determine the optimal hierarchical tree nodes. Meanwhile, the second hierarchical tree proposed is based on the work of Punera and Ghosh [13] where their tree is based on the taxonomy of the instances. Both proposed trees use as base learner SVM, but we also propose the use of Multinomial NB.

The final experiment involves training a Neural Network with a single hidden layer to perform a binary classification task (directed only to the IMDB dataset). This network has two sequential dense layers, with input connecting to a 10-dimensional hidden layer with ReLU activation function, cascaded by another dense layer, connected to output with a sigmoid activation function for prediction. After analysis, it is found out that binary cross entropy with Adagrad optimizer works best for this use case. The model is then compiled with the aforementioned parameters [14]. It is then trained with 1 epoch and a batch size of 200, considering the time and memory constrains.

V. RESULTS

As part of the task of investigating the performance of our classification models, we invested time in implementing the 5-fold method to measure the accuracy. As A 5-fold cross-validation was performed on each of the models and experiments. For the case of the 20 newsgroups the performance was based directly from the prediction of the test subset. Meanwhile, the IMDB dataset performance was based on the mean of the cross-validation performance. The performance is resumed below in Fig. 7.

From the previous table it can be observed that the highest accuracy obtained for 20-newsgroup dataset was from applying the combined voting classifier model proposed as experiment 2. Nevertheless, the accuracy of Naive Bayes was very close to experiment 2, but there was a substantial less computational cost given the fact that it's required to train and test 5 models simultaneously. Thus, one can say that there isn't a significant advantage of using any of the experiments with respect to Naive Bayes. Meanwhile, experiment 1 showed that in fact there isn't an increase of using Adaboost with SVM and NB as weak learners, given the fact that these models are too complex to boost. On the hand, this experiment proved that, for this dataset Decision Trees are not a good base learner since the accuracy is less than 50%. Finally, experiment three showed that the "n-gram" algorithm if better for hierarchical classification than categorizing by taxonomy. Nevertheless, the accuracies were not significantly

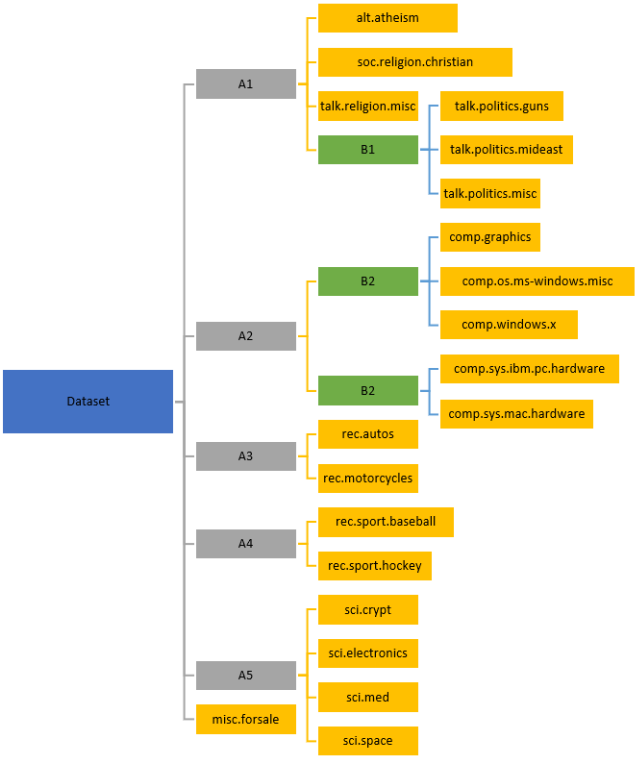

Fig. 5. First Hierarchical Tree. Taken from [12]

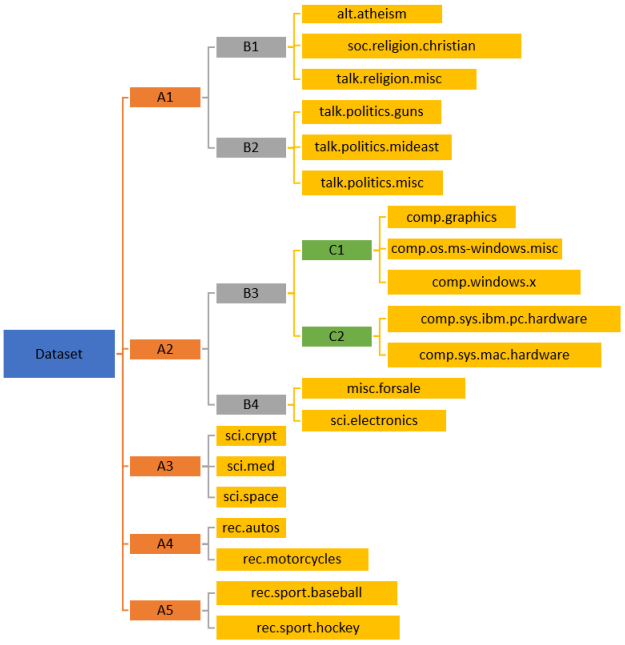

Fig. 6. Second Hierarchical Tree. Taken from [13]

	Model	Performance 20 Newsgroups	Performance IMDB
Basic Models	Logistic Regression	69.1052%	90.6120%
	Decision Tree	40.7594%	72.0560%
	SVM	69.8088%	**90.8280%**
	Adaboost (Decision Tree)	43.8263%	75.5240%
	Random Forest	62.5199%	85.0800%
	Multinomial NB	**70.1009%**	87.4600%
Experiment #1	Adaboost (SVM)	69.8088%	85.6680%
	Adaboost (Multinomial NB)	61.2985%	86.3000%
Experiment #2	Combined Classifiers	**70.3797%**	88.3760%
Experiment #3	First Hierarchical Tree (Multinomial NB)	67.6447%	N/A
	First Hierarchical Tree (SVM)	66.4365%	N/A
	First Hierarchical Tree (Logistic Regression)	66.0913%	N/A
	Second Hierarchical Tree (Multinomial NB)	48.8848%	N/A
	Second Hierarchical Tree (SVM)	46.6543%	N/A
	Second Hierarchical Tree (Logistic Regression)	46.4286%	N/A
Experiment 4	Neural Network	N/A	**90.800%**

Fig. 7. Test Results for Both Datasets

better than other models. As for the IMDB dataset, there was a tie for the best accuracy between SVM and Neural Network, but in terms of computational cost, the winner was SVM. We can see for experiment 1 that Adaboost didn't present an increase for SVM but it did present a better performance for NB (at the expense of computational cost). On the other hand, experiment 2 didn't present an increase in performance with respect other models.

VI. DISCUSSION AND CONCLUSIONS

The main takeaway from this presented work is that the process of text classification is performed via vectorization. This leads to a highly sparse and high dimensional matrix. The best models for this type of datasets resulted to be SVM, Naïve Bayes and Logistic Regression. On the other hand, the proposed experiments didn't bring better results, except for Neural Networks. For future work, a good approach would be research on more complex data cleaning and preprocessing methods specifically for text classification.

STATEMENT OF CONTRIBUTIONS

Luis worked on the data analysis and modelling necessary for the dataset 1 (20 News group dataset) while Manoj worked on the data analysis and modelling for the dataset 2 (Imdb Reviews dataset). Luis conceived and implemented the necessary experiments for dataset 1 and Manoj implemented the experiments for dataset 2. Israel worked on the report. Throughout the process, we helped each other and it was the collective effort that made this project successful

REFERENCES

[1] C., Crawford. 20 Newsgroups. Kaggle. 2011. Available : https://www.kaggle.com/crawford/20-newsgroups [Accessed March/02/2020]

[2] A. L., Maas, et al. Learning Word Vectors for Sentiment Analysis. Association for Computational Linguistics. Pgs. 142-150. 2011. Available: http://www.aclweb.org/anthology/P11-1015 [Accessed March/02/2020]

[3] F. Pedregosa, et. al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. Vol. 12, p. 2825-2830. 2011.

[4] S. Li, "Multi-Class Text Classification with Scikit-Learn," Towards Data Science, 19 February 2018. [Online]. Available:https://towardsdatascience.com/multi-class-text-classification-with-scikit-learn-12f1e60e0a9f. [Accessed 10 March 2020].

[5] S. Bansal, "A Comprehensive Guide to Understand and Implement Text Classification in Python," Analytics Vidhya, 23 April 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/. [Accessed 11 March 2020].

[6] H. H. Malik, D. Fradkin and F. Moerchen, "Single pass text classification by direct feature weighting," Knowledge and Information Systems, vol. 28, no. 1, pp. 79-98, 2011.

[7] M. Li, E. Ch'ng, A. Y. L. Chong and S. See, "Multi-class Twitter sentiment classification with emojis," Industrial Management & Data Systems, vol. 118, no. 9, pp. 1804-1820, 2018.

[8] Ting, K. M., & Zheng, Z. (2003). A Study of Adaboost with Naive Bayesian Classifiers: Weakness and Improvement. Computational Intelligence, Volume 19, Number 2.

[9] Li, X., Wang, & L., Sung, E. (2007). AdaBoost with SVM-based Component Classifiers. Science Direct, ELSEVIER. Engineering Applications of Artificial Intelligence.

[10] Saha, S., & Ekbal, A. (2013). Combining multiple classifiers using vote based classifier ensemble technique for named entity recognition. Data & Knowledge Engineering, 85, 15-39. doi:10.1016/j.datak.2012.06.003

[11] Ying, C., & Run-Ying. (2011). Novel top-down methodsfor Hierarchical Text Classification. ELSEVIER. SciVerse ScienceDirect. 2011 International Conference on Advances in Engineering.

[12] Graovac, J., Kovacevic, J., & Pavlovic-Lazetic, G. (2017) Hierarchical vs. Flat N-Gram-Based Text Categorization: Can We Do Better? Computer Science and Information Systems 14(1):103–121. DOI: 10.2298/CSIS151017030G

[13] Punera, K., & Ghosh, J. (2008). Enhanced Hierarchical Classification via Isotonic Smoothing. Data Mining – Algorithms.

[14] NA. (ND). Model class API. Keras Documentation. Available: https://keras.io/models/model/ [Accessed March/10/2020]