COMP30027 Report

Exploring ML Model Biases through Language Sentiment Classification

1. Introduction

Natural language processing is a field of science that has been everchanging. Its seemingly untouched nature brings more and more curious minds into the field, with more discoveries made every single day, making the most impossible tasks possible. One task remains a fundamental one to NLP, Text Classification. Sounding the most benign, contrary to its implications; it has made its way to the important parts of applications we use everyday; from spam filters to recommender systems used by virtually any eCommerce website.

In this report we will explore the effectiveness of two different classification models, Naïve Bayes and Decision Trees, in classifying language sentiment.

2. Background

The approaches taken in this investigation is made in order to show how model accuracy is affected by different models, biases, and their hyperparameters.

Naïve Bayes models are considered to be a strong baseline for text classification. They are relatively fast, easy to train, and substantially easy to understand. A journal from MIT, has found that despite its 'naïve' assumptions and systemic issues that the model is prone to having, simple corrections can produce considerably effective models, "*competitive with state-of-the-art text classification algorithms such as the Support Vector Machine."* (Rennie, et al., 2003). Naïve Bayes algorithms responds variable to different types of text, and benefits from simple alterations based on the applications. For example, the multi-variate Bernoulli strategies are effective with small vocabulary sizes, whereas multinomial performs well with larger sizes. (McCallum & Nigam, n.d.)

Decision Tree algorithms are another favourite among other popular ML learners, as it is very intuitive and simple to implement. They are great baselines for many different applications. Certain kinds of decision trees have been shown to outperform certain instances of Naïve Bayes. (Su & Zhang, 2006). We will further see how the model & its biases will compares to NB in this report.

The datasets has been kindly provided by Mukherjee et al. and Rayana & Akoglu.

3. Method & Results

The process used to construct the machine learning models consisted of two parts; determining the input data to train & test the models and selecting the hyperparameters relevant to each model.

There were two different types of encoding for the input data, doc2vec and count vectorizer. The doc2vec utilizes feature extraction methods, pruning irrelevant features, whereas Count Vectorizer preserves all.

Model decisions here are made by considering the accuracy of each model, using a **5-fold cross-validation**, a common statistical evaluation method for ML models. It splits the data into five partitions, selecting one partition as a test data, and using the rest for training; this is repeated with the remaining four partitions. The mean accuracy is the metric used to evaluate the models.

Note: this section highlights the

methodology and results; the analysis on observations made here will be discussed further in section 4.

3.1 Constructing the Naïve Bayes Model

To construct the Naïve Bayes model, we need to choose an algorithm type, and the specific data encoding to train the models.

3.1.1 Selecting Algorithm Type

There are different types of Naïve Bayes models that can be used for our text classification task. In this report we are considering three popular models of NB: **Gaussian, Multinomial, and Complement.** GaussianNB assumes a normal distribution for the likelihood of the features, whereas Multinomial assumes a multinomial distribution, and Complement is another implementation of MultinomialNB specifically suited for imbalanced data sets (Rennie, et al., 2003).

NB Algorithm	Accuracy(%)	σ^2
GaussianNB	38.51	3.04e-5
MultinomialNB	83.74	1.49e-5
ComplementNB	82.26	4.61e-6

Table 1 – Cross-validation scores of different NB algorithms

In table 1, you can clearly see the difference in the accuracies between the three Naïve Bayes algorithms. Despite not having the least variance, the most accurate model for this dataset is **MultinomialNB.**

3.1.2 Selecting Data Encoding

A comparison between cross-validation scores using two of the different encodings was used in choosing the best input data to use to train our model.

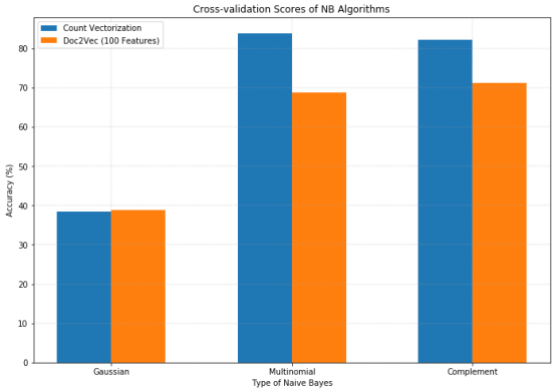

Figure 1 - Cross-validation of different NB algorithms.

As seen above, using data encoded with doc2vec significantly decreases the accuracy of Multinomial and Complement NB. Hence, we will be using Count Vectorizer encoded data to train our NB model.

3.2 Constructing a Decision Tree Model

In constructing our decision tree model, we selected effective combinations of hyperparameters, and the most suited data encoding strategy model.

3.2.1 Selecting Hyperparameters

The process of selecting optimal hyperparameters involved using grid search with cross validation. In essence, it is a brute force approach that tests different combinations of parameters using cross-validation, and selecting one with the highest accuracy. (Dangeti, 2017)

After running 27 different combinations of hyperparameters, the best parameter combination is shown below:

Hyperparameter	Value/Type
Impurity Criterion	Gini
Split Function	Best
MinSamples/split	1
MinSamples/leaf	2

Table 2 – Best Combination of DT

Hyperparameters found by GridSearch

2.1.2 Selecting Data Encoding

Considering the sensitivity of the DT model to high-dimensional data, doc2vec was used to extract the most relevant features. The results below suggests that we use the 100-feature extraction for optimum accuracy.

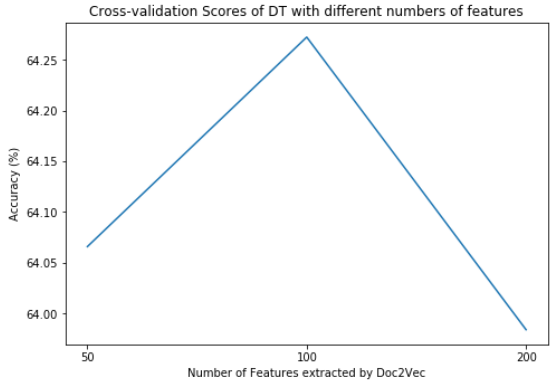

Figure 2 - Cross-validation accuracy scores of DT with varying numbers of features.

4. Analysis

4.1 Discussion of observations made on NB

We have seen that there were disparities in the accuracy between the NB models. The low accuracy of GaussianNB is mainly attributed to the 'naïve' assumption of the data being normally distributed. It is intuitively easy to imagine that the vocabulary used by reviewers are not normally distributed; hence, making this assumption was detrimental to the model. The closeness in accuracy between the other two NB models are most likely caused by the similar assumptions of multinomial distribution in the data.

Furthermore, in figure 1, NB showed improvements when using count vectorizer encoded data. NB is a resilient model able to handle large number of features, handling irrelevant features well, minimally affecting the overall results. The count vectorizer encoded data has a greater number of features, giving the model more information without substantially diminishing the accuracy of the model. Meanwhile, doc2vec's removal of 'irrelevant' features may have removed useful information to the classifier.

4.2 Discussion of observations made on Decision Tree

The DT's parameter selection was sometimes limited, for example, the other option for split function, was random, which did not consider the gain of information during a split.

MinSamples per split and leaf were best kept small, possibly because many reviews of the same category tend to use wide range of vocabulary, reducing the number of instances with similar composition of words, hence, increasing the minimum samples diminished the depth of the tree, consequently reducing the accuracy of decision tree.

Figure 2 also showed that using different number of features in the input data affected the effectiveness of the model. Intuitively, more information can be retrieved from additional features, hence 100 features yields higher accuracy than 50 features. However, as we increase it to 200 features, the model struggles to scale and negatively affected its accuracy. DTs are vulnerable to irrelevant features; the increasing number of features also encourage overfitting in the model, which ultimately decreases its accuracy.

4.3 Comparing NB and Decision Tree Models

From the results of the models that we have created, MultinomialNB have produced the most accurate predictions.

ML Model	Accuracy (%)
Decision Tree	64.27
MultinomialNB	83.74

Table 3 – 5-fold cross-validation accuracy scores of Decision Tree vs MultinomialNB.

The strengths of NB model can really be seen in the task at hand. Its robustness to high-dimensional data and extraneous features produces relatively accurate models. Whereas the effectiveness of DTs has been adversely affected by our sparse and feature rich dataset.

However, NB models are 'un-tunable' for the most part, the tuning are reserved in the pre-processing stage such as during data encoding. In contrast to decision trees, where there can be many decisions to be made for its hyperparameters (though difficult to tune).

Between the two models, we see how models can suffer greatly when datasets work against their biases & systematically flawed assumptions, as shown with GaussianNB and Decision Trees.

5. Conclusion

From our results & analysis, we have seen how certain models have their own strengths and blind-spots, and how they are significantly impacted by the input dataset itself.

In conclusion, it is important to stress the importance of understanding a ML model's assumptions and weaknesses, as they are critical in making sensible decisions when creating effective models.

6. References

Dangeti, P., 2017. *Statistics for Machine Learning.* s.l.:O'REILLY.

McCallum, A. & Nigam, K., n.d. A Comparison of Event Models for Naive Bayes Text Classification.

Rennie, J. D. M., Shih, L., Teevan, J. & Karger, D. D., 2003. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. *MIT,* pp. 616-23.

Su, J. & Zhang, H., 2006. *A Fast Decision Tree Learning Algorithm.* New Brunswick: University of New Brunswick.

Mukherjee, A., Venkataraman, V., Liu, B. & Glance, N. What Yelp fake review filter might be doing? 7th International AAAI Conference on Weblogs and Social Media, 2013.

Rayana, S. & Akoglu, L. Collective opinion spam detection: Bridging review networks and metadata. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015. 985-994.