Sentiment Analysis on social media: Comparison of the Efficient of the Models

by

Name: Chia-Yu Chen
Student-Nr.: S158800

Name: Tomasz Tadeusz Rogoz
Student-Nr.: S136267

Exam: Natural Language Processing and Text Analytics (CDSCO1002E) - Oral exam based on written product (IC)

Study: MSc Business Administration and Data Science
Date: 28. May 2023
Pages: 15
Characters incl. spaces: 25,999
Dataset:
https://www.kaggle.com/datasets/kazanova/sentiment140?datasetId=2477

Sentiment Analysis on social media: Comparison of the Efficient of the Models

Authors

Chia-Yu Chen (158800)

Tomasz Tadeusz Rogoz (136267)

{chch22ag, taro19ad}@student.cbs.dk

Students - Copenhagen Business School

MSc. Business Administration and Data Science

Abstract

Nowadays, social media plays an essential role in expressing opinions. Organizations can gain valuable insights into the public's attitude towards specific topics by conducting sentiment analysis on the text collected from social media. In this research, we utilized the Naïve Bayes classifier, logistic regression classifier, and the pretrained Bidirectional Encoder Representations from Transformers (BERT) on the "Sentiment 140" dataset, which consists of tweets. We compared the efficiency of different models. The results show that the BERT model achieved an accuracy of 0.88, with the same F1 macro average score, while the Naïve Bayes classifier achieved an accuracy of 0.74 and the logistic regression classifier achieved an accuracy of 0.77. This finding justifies that the transformer neural network performs outperform traditional machine learning algorithms. However, the significant computational cost required for training the BERT model should also be considered.

Furthermore, in an attempt to reduce the computational cost, we experimented with removing certain preprocessing stages, such as text cleaning and lemmatization. As a result, the overall processing time decreased to less than 1 second in the Naïve Bayes classifier. However, in the logistic regression classifier, the simplified preprocessing led to an increase in the number of input features, thereby increasing the time required for training the model. Nevertheless, in this research, the simplification of preprocessing did not have a significant effect on the accuracy rate in both the Naïve Bayes and logistic regression models.

1. Introduction

As technology progresses, social media has become the most popular communication tool in recent years (Dridi & Recupero, 2019). People use social media to connect with their friends, express their daily feelings, and share their opinions on the products they have bought or their attitudes towards companies or societal issues. Among the various social media sites, Twitter is a widely accessible platform for opinion sharing in different domains, including social issues and attitudes towards specific organizations (Yoo et al., 2018). Twitter also provides the Twitter Search API, which allows collection of tweets related to products, companies, and advertisements through keyword searches. By conducting sentiment analysis on these tweets, companies, organizations, and governments can gain valuable insights into public attitudes towards their products, policies, and advertisements, thus improving the decision-making process.

Sentiment analysis is a field in natural language processing that focuses on analyzing the public's attitude towards an object, such as a product, policy, commercial, or social issue on the internet (Alhojely, 2016). This paper will employ the generative classifier (Naïve Bayes Classifier), discriminative classifier, and the deep learning neural network (BERT) for sentiment classification tasks, using 50,000 tweets as training data. The goal is to compare the efficiency of these models in short text sentiment classification by evaluating their accuracy and computational cost.

2. Relevant Works

Using machine learning techniques to perform sentiment analysis has been a long-time feature of the natural language processing field. As novel techniques appear it is useful to evaluate their performance in a broad range of contexts since the results of machine learning models in general depend not only on their sophistication but on the particularities of the task, such as the dataset's

size and number of features, details of pre-processing etc. Geetha & Renuka (2021) compared BERT's performance to earlier models on a topic-based sentiment analysis task. It demonstrated that BERT significantly outperformed its predecessors across all metrics. Additionally, the Sentiment140 dataset has been utilized for sentiment analysis in the past by researchers utilizing various techniques from Naïve Bayes and Ridge Classification through Support Vector Machines up to Neural Networks. Based on a survey of sentiment analysis papers (Tan et al., 2023) their results ranged from an accuracy of 77% to 90% on the dataset in question. Using BERT on that dataset was not part of that survey so it is worthwhile to compare its results as well as check whether our implementation of Naïve Bayes and Logistic Regression performs similarly to previous ones. Our paper will aim to determine whether its advantage is maintained on the simpler task of binary classification of general sentiment.

3. Conceptual Framework

1.1 Model Comparison

3.1.1 Data Preparation

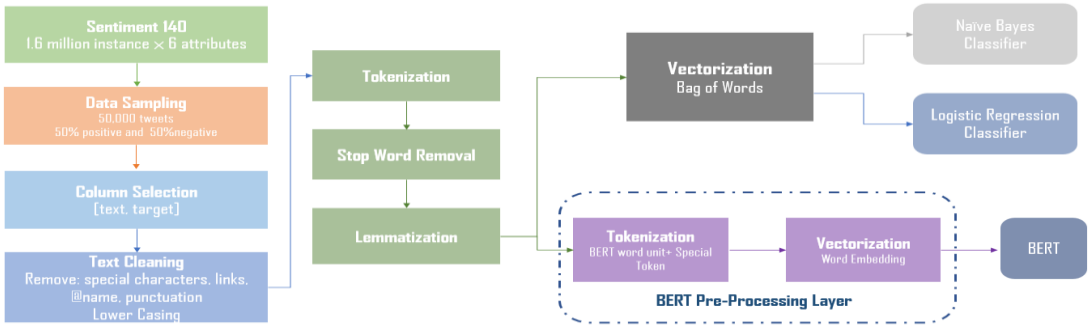

Figure 1 The Standard Pre-processing Process in this Paper

Since our data is gathered from the social media platform Twitter, the language is casual and unstructured. In order to transform the text into a format that the algorithm can process and learn the patterns for sentiment classification, a sequence of data preprocessing operations is necessary. The data preparation process can be roughly divided into five different steps: data sampling, column selection (which involves extracting the necessary attributes), text cleaning,

tokenization, and normalization, followed by vectorization (refer to Figure 1).

Due to the different input requirements of each model, the pre-processing methods will vary slightly in the tokenization and vectorization stages.

3.1.2 Training Strategy

This research will train various models, including the generative classifier (such as Naïve Bayes classifier), discriminative classifier (such as logistic regression classifier), and transformer-based architecture (such as pre-trained BERT model). Due to the different requirements for input data, the pre-processing methods for the models will differ slightly in the text tokenization and vectorization stages. However, the training set, test set, and the text cleaning process used for different models will be the same to ensure fairness in the comparison.

1.2 Computational Cost Reduction Experiment

We also analyzed whether we could save computational costs by simplifying the pre-processing process, or if the increased training time would offset the time saved. In this experiment, we skipped the text cleaning and lemmatization processes, and only focused on essential pre-processing steps such as column selection, tokenization, and vectorization to transform the data into a vector format. The transformed data was then fed into the Naïve Bayes classifier and logistic regression classifier to investigate the effects on accuracy and computational cost.

Figure 2 The processed of Simplified Pre-processing Process

4. Methodology

4.1 Data Description

The models used in this project were trained and tested on the “Sentiment 140” dataset, which

was created by Go et al. (2009) and is publicly available on Kaggle. Go et al. (2009) utilized the Twitter Search API to collect English tweets from Twitter and automatically annotated them. The dataset is provided in the form of CSV files, with each tweet represented as an individual row containing six different attributes: target, ids, date, flag, user, and text[1]. The dataset consists of 1.6 million tweets, with 50% of the instances labelled as positive and 50% labelled as negative. This balanced distribution ensures that the algorithm can learn patterns from both categories, thereby facilitating the evaluation of sentiment classification models.

4.2 Data preparation

4.2.1 Data Sampling

After conducting experiments on the Naïve Bayes and logistic regression models, it was observed that using the full dataset of 1.6 million tweets resulted in only a 3% improvement in accuracy compared to the sampled dataset of 50,000 tweets.[2] Therefore, it was decided to utilize the randomly sampled 50,000 tweets from the dataset while preserving the original class distribution. All models will be trained using this sampled dataset to ensure a fair comparison.

4.2.2 Column Selection

Since only the text column and the target column are required for performing sentiment analysis and the dataset contains 6 attributes, we have removed the columns other than text and target for further operations.

4.2.3 Text Cleaning

To prepare the text for the sentiment classification task, we have performed several pre-processing steps. Firstly, we removed special characters such as @, _, and <, as well as

[1] For the attribute description, please see Appendix 1: The Attribute Description

[2] For the comparison of accuracy of model trained on full dataset and sampled dataset, please see Appendix 2: The Accuracy Comparison Between Full dataset and 50,000 Sampled Dataset

punctuations from the text. Additionally, mentions (@username) and links were also eliminated, as we assumed these components do not significantly impact the sentiment of the text. Furthermore, all the text was converted to lowercase to ensure consistency throughout the dataset.

4.2.4 Text Tokenisation and Lemmatization

The 'nltk.word_tokenize' function is utilized to split the cleaned text into individual words. Stop words are then removed from the text, as they are generally considered to be insignificant in conveying sentiment. Subsequently, lemmatization is applied to reduce each word to its base form or lemma. This process helps reduce the sparsity of the feature space and simplifies the complexity of the text, making it easier for the algorithm to identify patterns by grouping words of the same type together. Additionally, lemmatization preserves the semantic information of the words, which can be beneficial for sentiment classification, as opposed to stemming, which may result in the loss of such information.

4.2.5 Vectorization

The pre-processed text is then converted into numeric values using different techniques. The first technique is the Bag of Words (BoW) method, which has been applied to the Naïve Bayes Classifier and Logistic Regression models. This technique transforms the text into a vector representation that contains the frequency of each word. However, it does not capture the contextual and sentiment information associated with each word. The reason for applying this method is that different models have different input requirements. For example, Naïve Bayes expects the input to be in the form of BoW vectors, while the pre-trained BERT model has its own vectorization technique.

4.2.6 Tokenisation and Vectorisation of BERT model

The pre-processing layer that handles transforming the data into a format suitable for the BERT

model was downloaded from TensorFlow Hub. It first tokenizes the text into word and sub-word units followed by adding special tokens to, for example, separate sentences or apply padding to sequences of varying lengths. Afterward, the text is transformed into vectors that represent its meaning based on the contexts of word occurrences during BERT's training, which is a process known as word embedding. Finally, positional embeddings are added that contain information about each word's position within the text.

4.3 Modelling Framework

4.3.1 Naïve Bayes Classifier

We chose the Naïve Bayes model as the baseline for our experiment because it is a basic yet powerful probabilistic classification algorithm that performs well on most text classification tasks, regardless of the size of the training data (Ayetiran & Adeyemo, 2012; Manning & Schutze, 1999). The Naïve Bayes classifier is based on Bayes' theorem, which assumes that the features are conditionally independent of each other. Under this assumption, the classifier calculates the posterior probability of each class label given the input instance's features and assigns the instance to the class with the highest posterior probability.[3]

In this research, we applied a smoothing method controlled by the 'Alpha' hyperparameter to handle unseen features. Looping through different alpha values was used to explore the number of occurrences to add for each word, such as one (Laplace smoothing) or less than one (Lindstone smoothing). The set of alpha parameter values we used was [0, 0.25, 0.5, 0.75, 1].

4.3.2 Logistic Regression

The second model we chose is the logistic regression model. Compared to Naïve Bayes, which is a classic example of a generative classifier, logistic regression is a well-known discriminative

[3] For the mathematical expression of Naïve Bayes model, please see Appendix 3: Naïve Bayes Classifier

classifier. It directly models the posterior probability and learns the direct mapping from instances to labels (Ng & Jordan, 2001).

Logistic regression is based on the sigmoid function, which maps the input instances to a value between 0 and 1. This value represents the predicted probability of the positive class. If the probability of the positive class exceeds the predefined threshold, the instance is classified as positive. To fine-tune the model, we tested five different values for the 'C' parameter ([0.1, 0.3, 1, 5, 10]), which controls regularization. Smaller values result in stronger regularization.

4.3.3 Bidirectional Encoder Representations from Transformers (BERT)

We also trained our data on the pre-trained BERT model, which is proposed by Vaswani et al. (2017). The model is an attention-mechanism based bi-directional transformer which learned to predict randomly masked (missing) words in a text corpus. Through that process it was able to acquire the ability to transform words into vector embeddings that preserve meaning based on the contexts of their occurrences within the training corpus as well as their importance within each context based both on previous and subsequent words. All that is required to fine tune the model on downstream tasks is to "simply plug in the task-specific inputs and outputs into BERT and finetune all the parameters end-to-end"(Devlin et al., 2018).

4.4 Performance Evaluation

While the accuracy rate provides insights into the proportion of correctly classified instances in the test set, it can overlook scenarios where all predictions are biased towards one class. In the case of our tweet sentiment analysis task, which is a binary classification problem, we utilize the F1 score[4] as the primary performance indicator for our models.

The F1 score is the harmonic mean of precision and recall, considering the trade-off between

[4] For the calculation of F1 score, please see Appendix 4: The Equations of Performance Metric

the two metrics. By using the F1 score, we can address the possibility of selecting a model with high accuracy but a high number of false positives or false negatives. The F1 score will be high only when both precision and recall rates are high, thereby providing a balanced evaluation metric. In addition to reporting the F1 score for each class, we also include the macro-averaged F1 score. By considering the macro-averaged F1 score, we ensure a comprehensive assessment of model performance.

5. Results

5.1 Model Comparison

		F1 score	*Precision*	*Recall*	*Accuracy*	*F1 score (macro)*
Naïve Bayes	Positive	0.73	0.77	0.69	0.74	0.74
	Negative	0.76	0.72	0.80		
Logistic Regression	Positive	0.77	0.76	0.78	0.77	0.77
	Negative	0.76	0.78	0.75		
BERT	**Positive**	**0.87**	**0.91**	**0.83**	**0.88**	**0.88**
	Negative	**0.88**	**0.85**	**0.92**		

Table 1 The performance of different models

To ensure a fair comparison, all models aside from BERT (training time constraints) were fine-tuned to optimize their performance. They were evaluated on the same test set consisting of 15,000 instances, with 7,512 positive tweets and 7,488 negative tweets. The results for F1 score, precision, recall, and accuracy are presented in Table 1.

Naïve Bayes, being the simplest generative model, achieved an F1 score (macro-averaged) of 0.74, which is the same as its accuracy. Logistic regression, as a representative of discriminative models, performed 0.03 better than Naïve Bayes in terms of both F1 scores and accuracy. BERT, with its more complex architecture, achieved an accuracy rate and F1 score of 0.88, which is

11% better than logistic regression, the second-best performing model in the experiment. Overall, model performance showed a strong correlation with model complexity.

When examining the performance on individual classes, both logistic regression and BERT demonstrated a difference of only 0.01 in F1 scores between positive and negative tweet classifications. However, in the Naïve Bayes model, the F1 score for negative tweet classification was 0.03 higher than that of positive tweet classification. To determine whether this 0.03 difference is a general characteristic of the Naïve Bayes classifier or specific to this test set, further analysis is required.

5.1.1 Computational Complexity Analysis

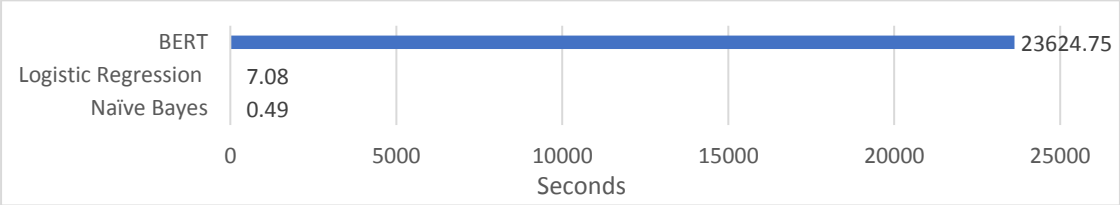

Figure 3 The Computational Cost of Different Models

The running time is also a critical factor to consider when analyzing the efficiency of the models. Sometimes, a significant computational investment may only lead to a slight increase in accuracy. To ensure a fair comparison, all models were trained on the Intel Xeon Gold 6130 v32 processor on Ucloud.

Since the data used to train these three models underwent the same pre-processing steps, the computational complexity analysis focuses on the difference in pre-processing and the main training process. For logistic regression and the Naïve Bayes classifier, the time indicated in the figure includes vectorization and training time. In the case of the BERT model, the computational cost encompasses the pre-processing layer as well as the main training time.

As shown in Figure 3, the computation cost of BERT, which involves multilayer transformer

neural networks, is approximately 6.5 hours[5]. In contrast, both Naïve Bayes and logistic regression require less than one minute to complete their computations.

5.2 Computational Cost Reduction Experiment

In the second experiment, we selected two models as the experimental set: the Naïve Bayes model and the logistic regression model. Each model was trained using two sets of training data. One set underwent essential pre-processing operations, while the other set underwent additional text cleaning and normalization. To compare the impact of different pre-processing approaches, we divided the total processing time into pre-processing time and training time. The simplified pre-processing approach reduces the time required for pre-processing but may increase the time spent on training. By analyzing and comparing the pre-processing time and training time separately, we can gain further insights into the efficiency and effectiveness of the models under different pre-processing conditions.

5.2.1 Naïve Bayes Classifier

Naïve Bayes	Accuracy	F1 Score (Macro Avg)
Standard Pre-processing	0.74	0.74
Simplified Pre-processing	0.75	0.75

Table 2 The Accuracy rate and F1 Score of Simplified Pre-Processed Naïve Bayes Model

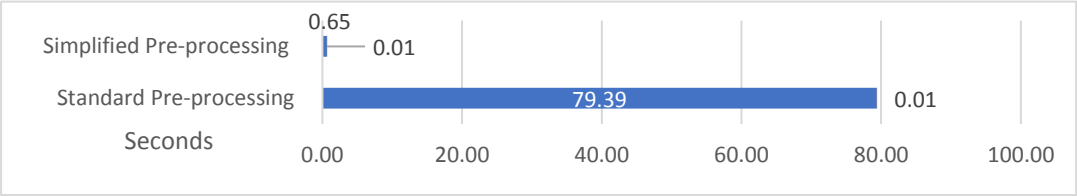

Figure 4 The Total Computational Cost of Naïve Classifier

[5] Due to the technical issues, the BERT training has been interrupted and only the training time of the last epoch has been recorded. Therefore, the 6.5 hours is estimated by the training time of last epoch times 5.

Since the Naïve Bayes model scales well as the number of input features increases, conducting the model with only simplified pre-processing allows the entire training process to be completed in less than one second. Moreover, in this specific dataset, the Naïve Bayes classifier surprisingly exhibits a slightly higher accuracy rate with the simplified pre-processing model. These findings suggest that the Naïve Bayes model is efficient when trained with a simplified pre-processing approach. The reduced pre-processing time enables faster training, while still achieving comparable or slightly better accuracy compared to the more complex pre-processing approach. It is essential to consider the specific dataset's characteristics and requirements when selecting and optimizing the pre-processing steps for a given model.

5.2.2 Logistic Regression

Logistic Regression	Accuracy	F1 score (Macro Avg)
Standard Pre-processing	0.77	0.77
Simplified Pre-processing	0.77	0.77

Table 3 The Accuracy rate and F1 Score of Simplified Pre-Processed Naïve Bayes Model

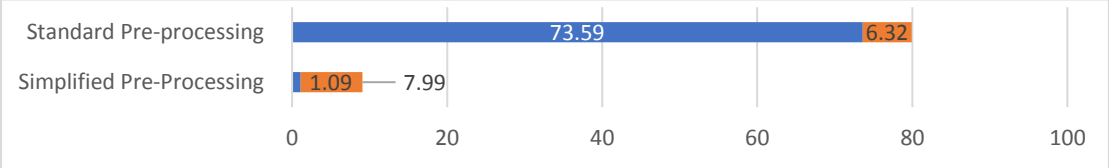

Figure 5 The Total Computational Cost of Logistic Regression Classifier

In logistic regression, the training time is highly positively correlated with the hyperparameter C. As the value of C increases, the algorithm tends to fit the training data more closely, requiring more iterations to converge. Fortunately, after grid search fine-tuning, both the optimized simplified pre-processing model and the standard pre-processing model have the same value of C1 (C=0.1).[6] Therefore, this comparison is based on models with the same hyperparameter but

[6] For the detail of fine-tuning result, please check Appendix 5: The Accuracy Result of Logistic

differently processed input data.

According to Table 2, the simplification of the pre-processing process does not significantly affect the accuracy rate and F1 score. However, in terms of computational cost, the total training time of the simplified pre-processed model is only 10% of the full pre-processed model. This is because the pre-processing time for 50,000 tweets significantly outweighs the training time of the logistic regression model with a C value of 0.1. However, upon closer examination of the training time cost, it is observed that the training time increases by roughly 25% in the model trained using the simplified pre-processed data.

6. Discussion

6.1 Comparison Between Different Models

Due to the assumption of the Naïve Bayes classifier, it is unable to capture the correlation between input features. In contrast, logistic regression can capture feature interactions by assigning weights to each feature. Furthermore, as a generative classifier, Naïve Bayes classifies instances based on probabilities, making it more sensitive to noise compared to discriminative algorithms, which can adjust the impact of different features. These reasons may explain why logistic regression outperforms Naïve Bayes in sentiment detection for social media content. However, the BERT model, being a transformer neural network, can preserve feature semantics, capture relationships between previous and subsequent features, and identify important features more accurately through self-attention layers. This leads to a significant improvement in model accuracy.

Nevertheless, as shown in Figure 3, the results demonstrate that although more complex algorithms can significantly increase accuracy, the computational cost should also be taken into

Regression trained by Standard Pre-processing Data and Simplified Pre-processing Data

consideration, particularly in online systems or systems that need to rapidly adapt to changing data.

6.2 The Effect of Pre-Processing Simplification on Training Time

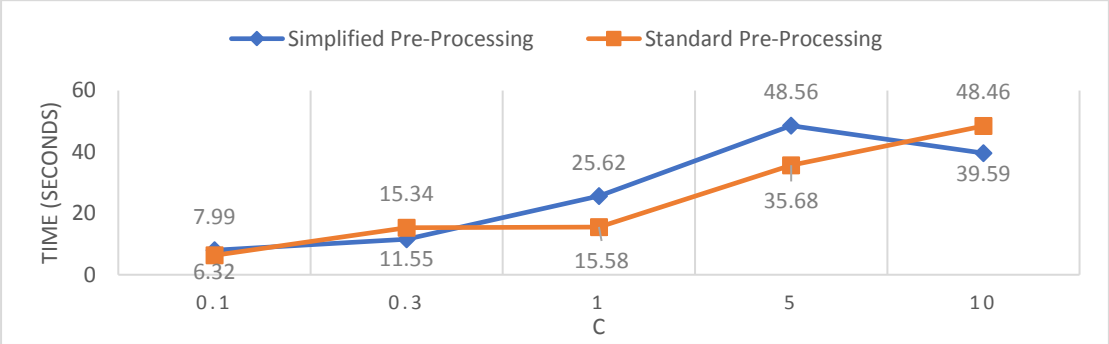

Figure 6 The Comparison of Training Time Between Logistic Regression Models Trained by Data Using Different Method

According to Figure 6, the training time of the model trained using simplified pre-processed training data is generally higher than the training time of the model trained using the standard pre-processing model. Moreover, the difference in training time increases from less than 2 seconds to 13 seconds as C value increases.

The exclusion of text cleaning and lemmatization in the simplified pre-processing process means that the training data might have more input features compared to the standard pre-processing process. This increase in input features can result in longer training times, particularly when the chosen algorithm is more sensitive to the number of input features. Therefore, the potential risk of longer training times may mitigate the reduction in computational cost generated by the simplified pre-processing process.

7. Conclusion and Further Works

In this research, we compared the efficiency of Naïve Bayes, logistic regression, and the BERT model on sentiment classification tasks using the Sentiment 140 dataset. In terms of accuracy,

the BERT model achieved an impressive 88% accuracy rate and F1 score, significantly outperforming the other models. This demonstrates that with sufficient training instances, a more complex architecture can capture complex relationships between different features, leading to better results. However, it is important to note that the BERT model also requires the largest computational power compared to logistic regression and Naïve Bayes, which can be trained in less than one minute. Therefore, when choosing a model, the trade-off between computational cost and accuracy should be considered.

In the pre-processing simplification experiment, simplifying the pre-processing did not have a significant effect on accuracy. For the Naïve Bayes model, which has very short training times regardless of the number of training instances, the simplified pre-processing reduced the total training time to less than one second. However, for the logistic regression model and other algorithms that are more sensitive to the number of input features, increasing the number of input features can lead to longer training times, which may offset the computational cost savings achieved through pre-processing simplification.

Due to computational constraints, we only used 50,000 tweets to train the BERT model. Although increasing the training set did not make a significant difference in Naïve Bayes and logistic regression, the BERT model, with its more complex architecture, may be able to learn more detailed patterns in the dataset. Therefore, we recommend increasing the amount of data and comparing the differences achieved with varying dataset sizes. Additionally, in this research, we chose to compare the BERT model with traditional classification techniques. However, to determine whether BERT outperforms most of the models available today, we also recommend comparing BERT with deep learning-based models, such as recurrent neural networks.

Reference

Alhojely, S. (2016). Sentiment Analysis and Opinion Mining: A Survey. *International Journal of Computer Applications*, *150*(6), 22–25. https://doi.org/10.5120/ijca2016911545

Ayetiran, E. F., & Adeyemo, A. B. (2012). A Data Mining-Based Response Model for Target Selection in Direct Marketing. *International Journal of Information Technology and Computer Science*, *4*(1), 9–18. https://doi.org/10.5815/ijitcs.2012.01.02

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.

Dridi, A., & Recupero, D. R. (2019). Leveraging semantics for sentiment polarity detection in social media. *International Journal of Machine Learning and Cybernetics*, *10*(8), 2045–2055. https://doi.org/10.1007/s13042-017-0727-z

Geetha, M., & Renuka, D. K. (2021). Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model. *International Journal of Intelligent Networks*, *2*, 64–69. https://doi.org/10.1016/j.ijin.2021.06.005

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. In *Sentiment 140* (No. CS224N). Stanford. http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip

Manning, C., & Schutze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

Ng, A. Y., & Jordan, M. I. (2001). On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. In *Neural Information Processing Systems* (Vol. 14, pp. 841–848). http://papers.nips.cc/paper/2020-on-discriminative-vs-generative-classifiers-a-comparison-of-logistic-regression-and-naive-bayes.pdf

Tan, K. L., Lee, C. P., & Lim, K. M. (2023). A Survey of Sentiment Analysis: Approaches, Datasets, and Future Research. *Applied Sciences*, *13*(7), 4550.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. In *arXiv (Cornell University)* (Vol. 30, pp. 5998–6008). Cornell University. https://arxiv.org/pdf/1706.03762v5

Yoo, S., Song, J., & Jeong, O. (2018). Social media contents based sentiment analysis and prediction system. *Expert Systems With Applications*, *105*, 102–111. https://doi.org/10.1016/j.eswa.2018.03.055

Appendix 1: The Attribute Description

target	The polarity of the tweet, 0 for negative, and 4 for positive.
ids	The unique identifier for each tweet.
date	The date of the tweet
flag	the query.
user	the user that tweeted the tweet
text	The content of the tweet

Table 4 Attributes in the sentiment 140 dataset

Appendix 2: The Accuracy Comparison Between Full dataset and 50,000 Sampled Dataset

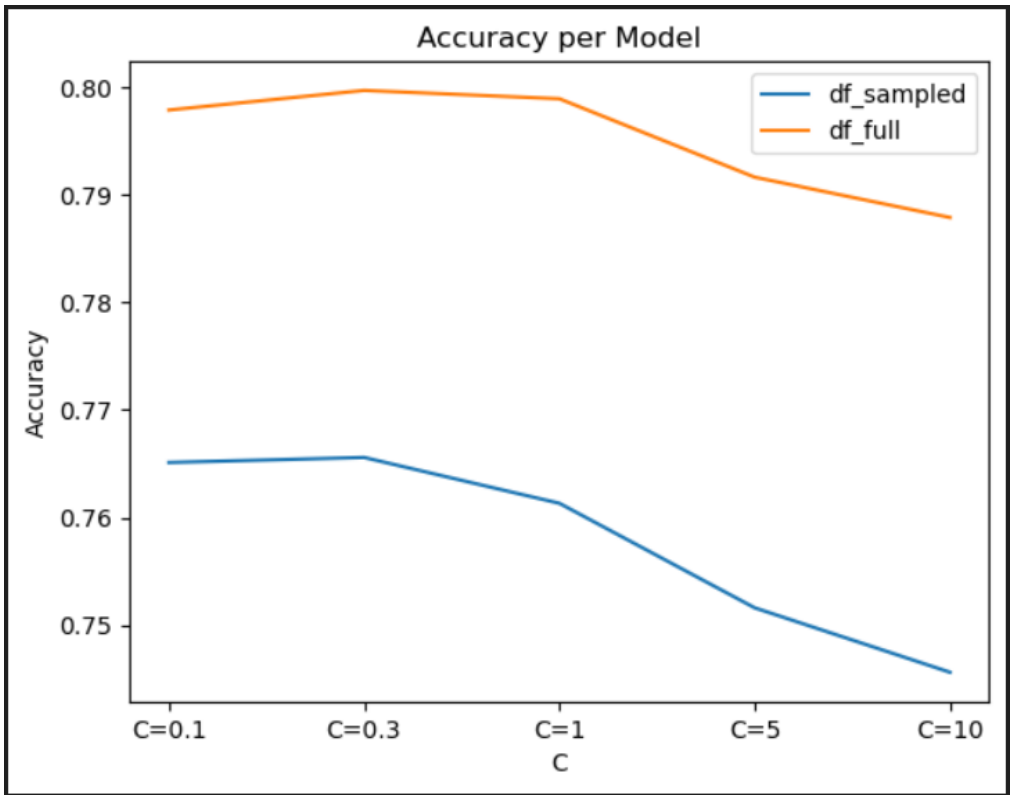

Figure 7 The Accuracy Rate Comparison Between Logistic Regression Model Trained by Full Dataset and Sampled Dataset. The graph shows that after fine-tuned there is only 3% accuracy different between

model trained by full dataset and sampled dataset.

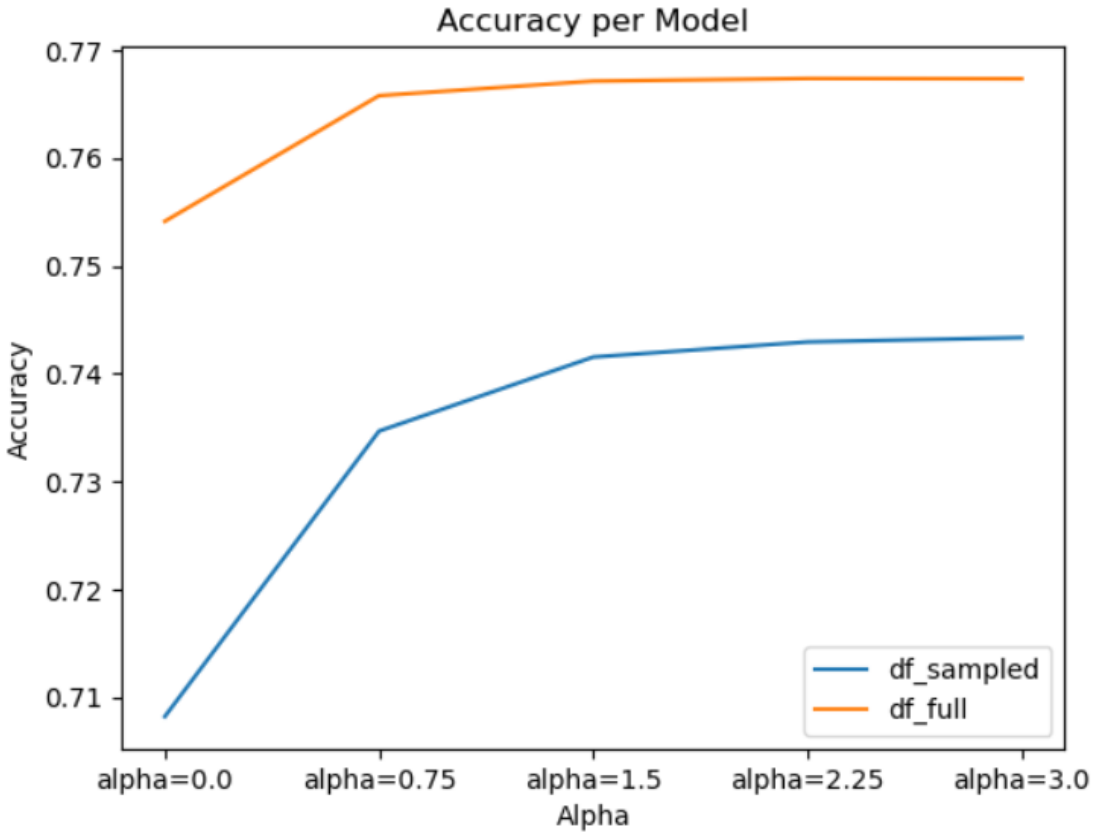

Figure 8 The Accuracy Rate Comparison Between Naive Bayes Model Trained by Full Dataset and Sampled Dataset. The graph shows that after fine-tuned there is only 3% accuracy different between model trained by full dataset and sampled dataset.

Appendix 3: Naïve Bayes Classifier

$$C_{NB} = argmax_{C_j \in C_{pos}, C_{neg}} P(C_j) \cdot \prod_{i \in position} P(f_i | C_j)$$

C_j: The j[th] of class from the set of possible class, in this case is 0 (C_{neg}) and 4 (C_{pos}).

f_i: The specific feature(lemma) in an input instance.

$P(C_j)$:The prior possibility[7] of j[th] class.

$P(f_i|C_j)$: The conditional probability of f_i belongs to the given classC_j

[7] The probability of the encountered feature belongs to j[th] class.

Appendix 4: The Equations of Performance Metric

$$Precision = \frac{\#\ of\ True\ Positive}{\#\ of\ True\ Positive + \#\ of\ False\ Positive}$$

$$Recall = \frac{\#\ of\ True\ Positive}{\#\ of\ True\ Positive + \#\ of\ False\ Negative}$$

$$F1\ score = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Appendix 5: The Accuracy Result of Logistic Regression trained by Standard Pre-processing Data and Simplified Pre-processing Data

C value	Accuracy	F1 score (macro average)
0.1	**0.77**	**0.77**
0.3	**0.77**	**0.77**
1	0.76	0.76
5	0.75	0.75
10	0.75	0.75

Table 5 The Accuracy Result of Logistic Regression trained by Standard Pre-processing Data

C value	Accuracy	F1 score (macro average)
0.1	**0.77**	**0.77**
0.3	**0.77**	**0.77**
1	0.76	0.76
5	0.75	0.75
10	0.75	0.75

Table 6 The Accuracy Result of Logistic Regression trained by Simplified Pre-processing Data