CHAPTER I

INTRODUCTION

1.1 Background

Now we live in an era with a tremendous and extensive amount of data. It is recorded that 59.5% of the world's 7.83 billion population are internet users, and 53.6% are active social media users [1]. Most of the data is unstructured, such as text, photos, videos, audio, images, et cetera.

Text mining or text data mining refers to the process of obtaining meaningful patterns and knowledge from unstructured data such as text [2]. One of the functions of text mining is sentiment analysis, which is finding the meaning and emotions of a text document.

Text documents obtained from social media, websites, et cetera., are derived from the typing of human fingers, making it possible to make typographical errors. An example is an abbreviated word, A Kaklauskas adding in [3]; abbreviations are one of the biggest problems because it will change the semantics of the word itself.

On the other hand, typographical errors make the machine think these words are different even though they have the same meaning. As a result, more and more data will be processed, and the data's dimensions will increase. Similar to what was stated by [4], one of the problems in text mining is the addition of vast data dimensions.

Existing algorithms are available to perform the classification, such as Rocchio, Boosting and Bagging, Logistic Regression, Naïve Bayes Classifier, KNN, SVM, Decision Tree, CRF, Random Forest, and Deep Learning [5]. Meanwhile, C. Aggarwal [6] groups the Naïve Bayes Classifier and Logistic Regression in the Probabilistic Classifier and adds a Neural Network to complete the algorithm mentioned in [5].

Rocchio and logistic regression are algorithms that are easy to implement; besides, the process and resources for computation are low, but Rocchio is not very good for multiclass datasets, and the algorithm is not very powerful.

KNN is one of the practical algorithms for text datasets and can handle multiclass datasets, but the computation is very high. Determining the optimal k value and finding the meaningful distance in the text dataset is the difficult part.

Decision Tree is a high-speed algorithm for prediction, but it is susceptible to the slightest disturbance in the data.

SVM has initially been designed for binary classification. To be applied to multiclass classification, it can use the one vs. rest, one vs. one, and the kernel method. Still, the use of the kernel has the disadvantage that it is difficult to choose an efficient kernel due to overfitting problems and training model problems.

Naïve Bayes works very well, especially in handling text data, multiclass data, and is easy to implement. Even Naive Bayes has a faster speed compared to other algorithms. Due to its practicality, computational efficiency, and good classification accuracy, the Naïve Bayes classifier is one of the most popular algorithms for classification. Witten also revealed that Naïve Bayes is speedy and accurate enough to handle the case of text documents [7].

This is supported by the results of research conducted by Baid et al. which is Naïve Bayes outperforming KNN and Random Forest performance by 81.45%. Dey et al also reveal that Naïve Bayes have better performance than KNN. Compared with SVM and Neural Network, the Mamgain et al. research shows that Naïve Bayes has better performance in the IIT dataset. Likewise with the decision tree which was outperformed by Naïve Bayes in the research of Al-Horaibi & Khan.

However, in the case of text mining, where the data is in the form of text, each word will be separated and become a separate and independent feature. Even though the meaning is the same, typographical errors will create new features, which increase the data dimensions. Besides, features data with typographical error is liable to the conditional probability to be 0, which can affect the posterior probability and undoubtedly affect the classification's accuracy.

Hence, phenomena and problems in typographical errors must be resolved. String matching can be a solution to correct typography by comparing two different strings. One of the existing algorithms performing string matching to

correct typographical errors is Levenshtein Distance used in research [8], [9], [10], [11], and [12]. Some others, like [13], had applied spell correction and got accuracy improvement.

1.2 Problems

1. Do typographical errors affect the classification model's performance?
2. How do typographical errors affect the classification model's performance?
3. Could typographical error correction be able to improve the model's performance?
4. What is the percentage improvement after correcting typographical errors?

1.3 Problems Limitation

1. The Levenshtein Distance algorithm uses the pyspellchecker.
2. English text is used for the data due to the limited languages dataset. The pyspellchecker only supports Latin characters, and only several languages can be used. Russian is the only available language for non-Latin characters.
3. Due to the limited dataset available for this research, the dataset got modified by adding typographical errors within.
4. This study does not discuss how the sentiment is in each dataset but discusses and analyses how the classification performance is.

1.4 Research Purpose

1. Find out whether typographical errors can affect the performance of the classification model.
2. Understand how typographical error affects classification model performance.
3. Find out if correcting typographical errors can improve classification model performance.
4. Discover how much improvement in model performance after typographical correction.

1.5 Research Benefits

1. Discover whether fixing typographical errors will improve classification model performance or not.
2. Enhanced insight and knowledge in the implementation of Naïve Bayes and the pyspellchecker for typographical correction and creating classifiers.

1.6 Research Methodology

1.6.1 Literature Review

The literature is sourced from scientific papers, journals, textbooks, and websites in this research. Scientific papers and journals are obtained from online service providers such as Google Scholar, IEEE Xplore, and Sciencedirect. The theoretical basis was taken from the textbook obtained from the pdfdrive website, and the dataset used in this research was obtained from the Kaggle and romisatriowahono.net website.

1.6.2 Problems Analysis

Naïve Bayes is an algorithm that calculates the probability and performs classification by finding the maximum value of the posterior probability. The posterior probability is obtained from calculating the likelihood probability with the prior probability. Typographical errors can cause the likelihood probability to be 0 because when the words are vectorized, the occurrence of this word will not exist; hence, it will make the likelihood value to 0, which will affect the posterior probability later.

1.6.3 System Design

The stages in this research begin with data collection continued with preprocessing the data, namely cleaning data with case-folding, removing regular expressions (symbols, punctuation, numbers), followed by typographical error correction, then tokenize the data, continued with removing the stopwords the last is stemming. After obtaining clean data, proceed with feature extraction using Bag of Words and enter the modeling stage with the Naïve Bayes Classifier with Laplace smoothing. The model that has been made will be evaluated for its performance with a Confusion Matrix and Cross Validation.

1.6.4 Implementation

Jupyter notebook will be the environment used in this research. The construction of the classification model will use the python programming language and the libraries that Python has provided, such as pandas, nltk, re, pyspellchecker, and sklearn.

1.6.5 Testing

Testing in this study will use several datasets shown in **Table 1.1** and the word frequency list used for the pyspellchecker shown in **Table 1.2** below.

Table 1.1 Dataset

Dataset	Source
Yelp	https://romisatriawahono.net/lecture/dm/romi-dataset.zip
Amazon	
Financial	https://www.kaggle.com/ankurzing/sentiment-analysis-for-financial-news?select=all-data.csv

Table 1.2 SpellChecker Word Frequency List

Word Frequency List	Source
Original	SpellChecker built-in list
Gwicks	http://www.gwicks.net/textlists/english3.zip
awesomeopensource	https://github.com/dwyl/english-words/blob/master/words_alpha.txt

1.6.6 Report Formulation

The model's performance created will be evaluated with a confusion matrix and cross-validation and compared to the model's performance before and after applying the typographic error correction.

1.7 Systematics Writing

CHAPTER I INTRODUCTION

Contains the background, problem formulation, problem limitation, research purpose, research benefits, research methods, and writing systematics.

CHAPTER II RELATED WORKS

This chapter describes the theories used and related to this research.

CHAPTER III RESEARCH METHODOLOGY

This chapter contains an analysis of the methods used to improve the quality of the Naïve Bayes Classifier.

CHAPTER IV IMPLEMENTATION AND DISCUSSION

This chapter contains a discussion of implementing the methods used and testing the results obtained.

CHAPTER V CONCLUSION

This closing chapter contains the conclusions obtained by the author through the previous chapters and answers the problem formulations in chapter 1, as well as suggestions for further research.

REFERENCES

This section contains a list of references that have been used in writing.