Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

Support Vector Machine and Naïve Bayes for Personality Classification Based on Social Media Posting Patterns

Bayu Seno Nugroho*, Warih Maharani

School of Informatics, Informatics, Telkom University, Bandung, Indonesia
Email: [1,*]baysen@student.telkomuniversity.ac.id, [2]wmaharani@telkomuniversity.ac.id
Correspondence Author Email: baysen@student.telkomuniversity.ac.id
Submitted: **06/12/2024**; Accepted: **12/12/2024**; Published**: 19/12/2024**

Abstract−This research investigates the use of Support Vector Machine (SVM) and Naive Bayes models to classify the personality traits based on the social media posting patterns. This study integrates textual features obtained from the Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) methods, and along with the feature expansion using the Linguistic Inquiry and Word Count (LIWC) tool, to assess their influence on accuracy Classification Personality characteristics were mapped from social media posts using the Big Five Inventory (BFI-44). The research findings show that the SVM model in which uses the TF-IDF + LIWC feature set, provides the best performance, and achieve 76.60% of accuracy on the base model with a linear kernel. In comparison to the Naive Bayes model performed best with the same feature set, achieving 59.57% accuracy with a smoothing parameter of 1xE-2. Although the oversampling improved recall and precision, the undersampling was found to have a negative effect on model performance. These findings highlight the benefits of combining TF-IDF and LIWC features which improve model effectiveness, with SVM producing the best overall results in personality classification from social media data.

Keywords: Support Vector Machine; Naive Bayes; Personality Classification; Social Media; Text Classification; BFI-44

1. INTRODUCTION

Personality can be identified as the mindset of a person, which determines the behavioral pattern, feelings, and attitude of a person, uniquely distinguishing an individual from others as a unique set of personality traits [1]. Various psychologists have developed psychological theories in order to predict personality. Some of them are Big Five Personality Traits, MBTI, Strengths Finder, and DISC Personality. Among them, Big Five Personality Traits is considered one of the most reliable models for classification because it categorizes personality traits into five broad categories: neuroticism, extraversion, openness, agreeableness, and conscientiousness [2][3]. Those aspects, being structured and quantified for understanding human behavior, have also found applications in different fields like personality classification from text data. Recent works within the domain of social media analytics show that text-based data, which correspond to posts on platforms such as X, are quite indicative of an individual's personality traits, as they actually reflect underlying emotions, attitudes, and behaviors in general [4][5][6].

Growing use of social media, especially X (formerly Twitter), has opened up unique opportunities for personality analysis, whereby the text within posts could become a rich source of behavioral signals. Considering the fact that personality traits (like extraversion and agreeableness) are conveyed in the usage of language-word choices, sentimental feel, and pattern of social interaction-it becomes feasible to increasingly employ machine learning models on these textual signals for making predictions regarding personality. The Big Five Inventory-44 (BFI-44) questionnaire, commonly used in psychological research, can be used to quantify the traits and match them to the corresponding patterns in social media data, as social media posts most often reflect personal experiences and states of emotion [5][6]. Recent studies have shown that text data from social media may serve as a very promising source for the inference of personality traits since it bears rich behavioral signals [7]. The increasing usage of platforms, such as X, which give users the ability to convey their thoughts in short forms of posts, is a very good setting for performing this kind of analysis.

Text classification is widely known to be performed by some machine learning methods such as NB and SVM; these are widely applied in different studies because of their effectiveness in managing text data and their results that have been well established already in several studies [8][9][10]. Naïve Bayes enjoys favorable considerations because of its simplicity and efficiency in handling probabilistic models. Several studies have proved that it reaches high accuracy in text classification tasks, even when the dataset is small. For example, in the study of Natasuwarna [10], it reached 89.86% accuracy with Naïve Bayes in personality classification, thus proving its potential in social media analysis. The performance of NB with small datasets is another important advantage; therefore, it is very popular in research studies where the sample size is small [11].

On the other hand, Support Vector Machine is a powerful technique when it comes to handling big and complex data with high accuracy. It is mainly powerful because it uses kernel functions, which enable it to map the data into higher-dimensional spaces so that it is able to handle non-linear data classification efficiently [12][13]. In this way, SVM has especially been suited for text classification problems where relationships among the features can be complex and nonlinear. For instance, in work by Fikriani et al. [14], personality classification has been made using an SVM on tweet data to arrive at an accuracy value of 86.88%. Widyadhana et al. [15], on the other hand, have illustrated the supremacy of Support Vector Machine over other methods while offering an average accuracy value of 96.43%. This work presents a study using Naïve Bayes and SVM in classifying personalities in social media posts made via X. In the following, the Big Five Inventory-44 (BFI-44) questionnaire is leveraged to collect personality data for

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

Indonesian users and tweets by analyzing their personality from such data. This present research will look into identifying the better method between the two Naïve Bayes or SVM methods in this context. Additionally, it identifies elements that may influence the outcome of both approaches. The data will be pre-processed, and then both models will be developed and evaluated regarding their performance in classifying personality traits based on textual data. Additionally, the research will emphasize those factors that might influence the success of these methods, such as the length of the posts, the frequency of tweets, or the linguistic styles used by the individuals.

It is expected that the results from this research will give further insights into the effectiveness of Naïve Bayes and SVM in personality classification using textual data from social media posts, hence helping in refining the existing models and contributing to the growing area of personality computing.

2. RESEARCH METHODOLOGY

2.1 Research Stages

In this research, a system was developed to classify personality traits based on social media posts using machine learning methods. The research stages are represented in the system flowchart design shown in Figure 1.

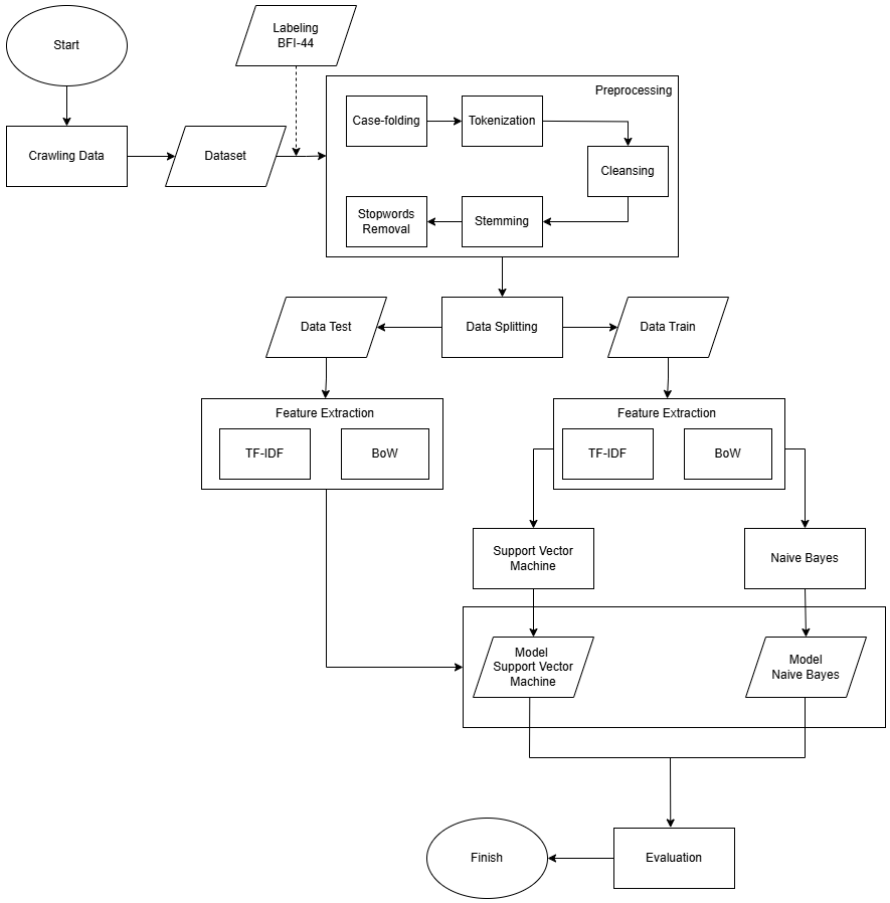

Figure 1. System Flowchart Design

Figure 1 presents a flowchart of system design, covering data collection, preprocessing, feature extraction, model training, and its evaluation. It begins by crawling texts from the social media platform X, wherein posts from people who allow their data to be used have been taken into account. The BFI-44 Big Five Inventory-44 then maps data from each dataset, labeled thereafter, into each one of the Big Five personality traits, including neuroticism, extraversion, openness, agreeableness, and conscientiousness.

The data preprocessing follows in line with the collection of data in the analysis. Here, preprocessing involves some series of case folding to lowercase, tokenization into individual words, cleansing-undesirable characters or symbols and incomplete data, removal of stopwords, and removal of very common and less essential words; stemming is used to get back roots. These steps ready the text data for good extraction of features.

Following preprocessing, data splitting splits the dataset into a training and testing set. These shall be used to develop the machine learning models, while the testing set is needed for performance evaluation of machine learning models. Each of the subsets then undergoes feature extraction, the end product of which will be representations in numerical forms for purposes of machine learning. In this paper, the two feature extraction techniques explored are TF-IDF of terms frequency-inverse document frequency and BoW.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

These extracted features are used to train two machine learning models, namely Support Vector Machine and Naïve Bayes. The Support Vector Machine uses a hyperplane-based approach with kernel functions for nonlinear data, whereas Naïve Bayes uses a probabilistic approach based on Bayes' theorem. For better performance of the Support Vector Machine model in text classification tasks, hyperparameter tuning is performed.

After training, both models will be tested using the testing dataset to see their performances. A few metrics that could be used in finding how well the personality traits are predicted by each model are: accuracy, precision, recall, and F1-score. Further, a comparison is made between the Support Vector Machine and Naïve Bayes models in analyzing which one yields better results for the dataset.

This is followed by comparing the results and developing overall analysis. From this, further guidance on the choice of the best model with regard to personality classification is gained. This study concludes by summarizing the findings of discussing implications for future research studies regarding personality trait analysis using machine learning.

2.2 Dataset

This paper is based on a dataset of 177,437 tweets collected from 233 unique users of X, a social network. A personality trait for each user is defined through the content of the tweets themselves, which have been categorized using the Big Five Inventory-44 (BFI-44). The dataset contains a number of important attributes. Username: It provides a unique username for every particular user that has posted at least one tweet, which again can be tracked and have his behavior classified based on this.

Cleaned Tweet: text of the tweet after the pre-processing that may contain tokenization, stopword removal and stemming amongst others, was done to the text to prepare it for analysis. Kecenderungan: personality trait label of each user based on his/her response upon the Big Five Inventory44 (BFI-44). The labels fall into five personality traits, to wit: ***Openness*** for 89 users, ***Agreeableness*** for 85 users, ***Neuroticism*** for 37 users, ***Conscientiousness*** for 17 users, and ***Extraversion*** for 5 users. The labels reflect the personality of each user and are very essential in training the machine learning models in this study. The following is a sample of the dataset used in the study presented in Table 1.

Table 1. Sample of Dataset

Username	Cleaned Tweet	Kecenderungan (target label)
a2lir	zsjl whoa weird victor valdes confucius lekat ..	Agreeableness
abdee	raya idul adha bold riders morowali distribusi..	Openness
achadianrani	ondel ondel ngejengkang kesandung speakernya g..	Extroversion
…	…	…
zyoslin	pas baca kayak lupa akhlak zoo gep mama akak t..	Neuroticism

2.3 Preprocessing

Preprocessing is one of the most important stages in this research, which deals with the preparation of the dataset for machine learning model training. The steps followed in preprocessing ensure that the raw data is clean, consistent, and suitable for feature extraction and model evaluation. Preprocessing was performed in the following steps:

a. Case Folding
   First step preprocessing is converting the whole text to lower case. It just makes sure that model would not differentiate between "Happy" and "happy.". Case folding was done in all tweets by a very simple function converting the text to lower case.
b. Cleaning
   Any text data may contain noise in the form of URLs, special characters, hashtags, mentions, and numbers. The cleaning function makes use of a regular expression in order to remove them. It actually cleans by removing URLs, special characters, numeric values, and unnecessary punctuation, retaining meaningful text content. This helps in reducing the noise in irrelevant information that may hamper the analysis.
c. Tokenization
   Cleaning was followed by the tokenization of the tweets into individual words using the word_tokenize function. Tokenization splits the text into words, or tokens, against which further cleaning can take place, like stopword removal and stemming.
d. Stopword Removal
   These would include general terms that would generally take no significant meaning, such as "the", "and", or "in". As such, words were removed with the help of a pre-defined Stopwords list. Besides standard stopping words, extra stop words could have been added within a person's custom stopword listing, such as informal vocabulary like "wkwkwk", "hahaha"., in bringing down the text dimension and focusing the analysis on word groupings that bear more informative text.
e. Normalization

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

There are colloquial or abbreviated forms some words can take in this dataset. Normalization standardized all the text using a colloquial Indonesian lexicon that mapped informal or nonstandard words to their standard forms. The normalization step hence allows variants of the same word to be treated as the same for better consistency of the dataset.

f. Stemming
It involves reducing words to base roots: For example, words such as "running," and "runner" became stemmed into the base form, "run." As indicated in Figure, the process of stemming simply normalizes words through some kind of library of stemmer, changing them to some form with a proper root, allowing this core meaning of words-ignoring variations-to be analyzed.

g. Language Detection
A language detection function is applied to ensure that, after this step, only the Indonesian tweets remain for subsequent analysis. This step retains only the tweets written in Indonesian, filtering in their languages for the study, since the focus of research will fall on Indonesian Language data.

h. Data Filtering
After cleaning and preprocessing, the dataset was filtered further by removing all rows with missing or empty tweets. This was done to ensure that valid, clean text data appeared in the dataset used for training and evaluation. Similarly, users with less than a set number of words were removed to ensure that there would be enough to train the models.

With these preprocessing steps, the dataset was ready for feature extraction to train a machine learning model. Save the final dataset for future analysis, and structure the preprocessed text in a cleaned form that could be used for classification tasks. This will ensure that the dataset is adequately prepared for training models with the ability to make efficient predictions of personality traits with the cleaned tweets.

2.4 Feature Extraction

Feature extraction is one of a very important step of transforming the raw text input into its numerical representation comprehensible to any machine learning algorithm. Hence, there have been adopted two feature extraction approaches, to transform preprocessed tweets in a quantitative manner: namely, the Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF). Both methods will be described according to different methods in converting text into structured characteristics for classification problems.

a. Bag of Words (BoW)
The model of the Bag of Words is a simple but effective method for feature extraction. It represents text data as a bag of individual words (or tokens) while ignoring syntax and word order, taking into consideration word frequency in a document. In BoW, each tweet is represented as a vector in which each element corresponds to one word from the overall vocabulary over all tweets. The value at each position denotes how many times that particular word comes in the tweet. Now, let us assume that we have a vocabulary: ["rain", "weather", "storm"]. So, with regard to a given tweet, "rain storm rain" - the BoW will be [2, 0, 1], by the counts of "rain", "weather", and "storm", respectively, in the corresponding tweet.
In BoW, the representation is sparse, meaning it has a lot of zeros, especially when the vocabulary size is big, since for each tweet only a small subset of words will appear. This forms one of the major limitations to BoW; however, it remains a popular and effective method for many text classification tasks.

Table 2. Example of BoW Feature Extraction

Tweet ID	*Word 1*	*Word 2*	*Word 3*	*Word 4*	*Word 5*	*…*
1	2	0	1	0	3	…
2	0	1	2	1	0	…
3	1	1	0	2	1	…
…	…	…	…	…	…	…

The above table represents each row as a tweet, and each column is the count of particular words (rain, weather, and storm) in that tweet. These feature vectors then generated by BoW act as an input to machine learning models.

b. Term Frequency-Inverse Document Frequency (TF-IDF)
TF-IDF is an enhancement for feature extraction which judges the importance of a word in a document against its appearance frequency in the entire set. TF-IDF grants more weight to words that appear frequently in one tweet yet are infrequent in all tweets, putting more emphasis on important terms and reducing the weight of commonly occurring terms with limited discriminative power. While Term Frequency considers the frequency of occurrence of a term in a single tweet, Inverse Document Frequency considers the rarity or commonality of a word across all tweets. terms that appear often in tweets carry a low IDF, and terms that appear in fewer tweets carry a high IDF. The TF-IDF score for a word in a tweet will be the product of the term frequency and the inverse document frequency, and the score tells the importance of each word from the perspective of the entire dataset. For example, suppose a word "rain" is very frequent in a tweet and barely appears in other tweets; this gets it a higher weight, whereas common words like "the" will have lower weights.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

Table 3. Example of TF-IDF Feature Extraction

Tweet ID	Word 1	Word 2	Word 3	Word 4	Word 5	…
1	0.56	0.00	0.42	0.00	0.69	…
2	0.00	0.45	0.61	0.39	0.00	…
3	0.33	0.45	0.00	0.56	0.33	…
…	…	…	…	…	…	…

By applying both BoW and TF-IDF for feature extraction, this research was able to capture different aspects of the textual data: BoW focuses on word frequency, while TF-IDF provides a more nuanced representation by adjusting for word importance. These feature vectors enable us to train classification models that can effectively predict user personality traits based on their tweets.

2.5 Feature Expansion

Feature expansion is one of the major steps in enhancing the performance of a machine learning model, especially when text data is considered. In this study, feature expansion was done using the Linguistic Inquiry and Word Count tool. The LIWC is a psychological text analysis tool that categorizes words into different psychological, emotional, and linguistic categories. Therefore, LIWC has been applied with the hope of enriching the feature set for the text data by extracting additional sentiment and psychological attributes that may help in enhancing the present classification task.

Features generated depend on a dictionary of words that are pre-grouped into categories such as emotional tone, social processes, and cognitive processes among other psychological aspects. In this problem, each tweet will be analyzed with the LIWC tool in order to generate an additional set of features to be used in machine learning models.

In this research, LIWC expands the feature space by adding these categorized word frequencies to the original textual features such as BoW and TF-IDF. It gives a better view on the underlying psychological and emotional characteristics of the text and helps the model for the tasks associated with personality classification or sentiment analysis. This will couple with some classic text representation methods, such as BoW and TF-IDF, provide more informative input for the classifier; hence, it would result in better performance in terms of accuracy for a model.

Table 4. Example of LIWC Feature Expansion

Tweet ID	***Funct***	***Article***	***Work***	***Achiev***	***Home***	**…**
1	-0.581217	-0.488783	-0.492269	-0.451321	-0.294414	…
2	-0.546151	-1.129493	-0.652176	-0.898800	-0.472746	…
3	-0.476020	-0.969316	4.761805	3.320293	0.121694	…
…	…	…	…	…	…	…

2.6 Data Sampling

Data sampling techniques are important aspects in data handling. Some data sets could have imbalance within classes; this would also mean fewer instances of one class compared to others. It makes the performance of these models suboptimal as these models usually get biased toward the majority class [16]. In this research, random oversampling and random undersampling are some of the data-sampling techniques used to handle the class imbalance problem so that all machine learning models will have a balanced set of data on which to train.

In this work, Random Oversampling was applied to personality traits with fewer instances, like Extraversion and Conscientiousness, to increase the number of users within these classes. It increased the number of users within these classes by simply duplicating the minority class samples, providing the model with more examples from each personality trait to learn better features for the model and make accurate predictions. However, this is an overfitting technique because it balances the dataset while possibly making the model just memorize the duplicated instances instead of generalizing from them.

On the other hand, Random Undersampling is the process of reducing instances of the majority class. This method removes random samples from the over-represented classes to make the dataset balanced. In this work, Random Undersampling has been performed on ***Openness*** and ***Agreeableness***-majority personality traits-reducing their instance numbers to the level of the minority classes. While Random Undersampling prevents the model from being biased toward the majority classes, it can lead to a loss of some valuable data, especially when too many samples are removed. This actually decreases the generalization capability of the model across the complete dataset. Hence, the amount of removed instances was considered so that it takes a good balance between not giving bias and keeping enough for performance.

Therefore, the current research used both techniques of Random Oversampling and Undersampling in order for machine learning models to train from a balanced dataset, thus minimizing bias and enhancing the model's general predictive performance.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

2.7 Splitting Data

Splitting of data simply refers to separating data into smaller blocks for both the training and testing of any model. Its purpose is to ensure, after being trained, one can always test the model and accurately score its performance. The split takes place in a way such that there will be only two categories involved: a test data and a train data. Test data are meant to evaluate model performance, ensuring correct prediction. Conversely, training data is used to train and develop the model on, which yields precise estimation of performance on new previously unknown data.

2.8 Support Vector Machine

The Support Vector Machine is one of the most famous supervised learning techniques for classification and regression problems. The basic idea of SVM includes finding the optimal hyperplane in a high-dimensional feature space that can maximize the separation between the various classes of data points. The hyperplane depends upon support vectors, which means those data points will be nearest to the hyperplane, thereby determining the position and orientation. [17].

Although it can't solve nonlinear decision boundaries straightforwardly, SVM can project input data onto higher dimensionality by so-called kernel tricks such that the classes could be separated by a hyperplane there. Some of the most popular ones in practice include linear, polynomial, RBF, and sigmoid. Nowadays, research into performance improvements is ongoing, especially those trying to construct new types of kernel functions according to special data distribution for precision improvement. Besides, some optimization approaches have been introduced to improve scalability, which makes SVM suitable for big datasets. [18]. Due to its robustness, ability to work in high-dimensional spaces, and suitability for both linear and nonlinear classification problems, SVM remains a popular choice. The separating hyperplane or decision boundary is a geometric approach to separate the various classes of data. The support vectors are the closest data points in the dataset to the said hyperplane and define margin. In SVM, this margin can be elaborated as distance of a hyperplane from its nearest support vector. The equation of the hyperplane as in SVM represented is given below:

$$f(x) = w^T x + b \tag{1}$$

Where $\boldsymbol{w}$ represents the weight vector, $\boldsymbol{x}$ is the feature vector of the training data, and $\boldsymbol{b}$ is the bias term, a scalar value.

2.9 Naive Bayes

Naive bayes is a classification algorithm that uses the Bayes' Theorem by making independence assumption of the features. It calculates the prior probabilities, likelihood, and posterior probability of a class in the input feature. It works well with huge datasets or text classifications [19]. The Naive bayes formula can be described by below:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)} \tag{2}$$

Where, *P(C|X)* is the posterior probability of class *C* given the feature vector *X*; *P(X|C)* is the likelihood of observing the feature vector *X* given the class *C*; *P(C)* is the prior probability of class *C*; and *P(X)* is the probability of the feature vector *X*. The Naive Bayes method assumes each feature is independent of the others, therefore it simplifies the computation *P(X|C)* simply as the product of individual probabilities of features. Therefore, it enhances the efficiency and scalability of the computation made in the algorithm [20].

Despite its simplicity and the strong independence assumption, Naive Bayes often performs at least as well as expected on text categorization, spam filtering, and sentiment analysis. The efficiency of this model in handling high-dimensional data makes it the first preference in many real-world applications where a quick, scalable solution is required.

3. RESULT AND DISCUSSION

Results regarding the personality classification of individuals based on their social media posts are presented and discussed, with the performance of the model in each stage highlighted. Various metrics are used to determine how well the results stand. The performance of Naïve Bayes versus the Support Vector Machine from this proposed data collection in Chapter 2 is depicted by showing the results.

3.1 Data Crawling and Preprocessing

The data for this experiment were crawled from X, formerly known as Twitter, by users who gave their consent that their posts be used for research. The labeling was made for each post in order to make them fit the five big personality traits, which include neuroticism, extraversion, openness, agreeableness, and conscientiousness using Big Five Inventory-44. The crawling succeeded in collecting 191.465 tweets from 233 users. The data preprocessing was applied effectively, including tokenization, stopword removal, stemming, and cleansing. After preprocessing, 177.437 tweets were available for feature extraction and model training.

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

3.2 Feature Extraction

The preprocessed data was transformed into numerical form for machine learning with the help of two techniques of feature extraction: TF-IDF and BoW. The features were extracted so that the text data may be represented in a way to allow the machine learning models to learn and classify personality traits effectively. The dataset generated 13.514 features using TF-IDF, with more weight assigned to words that were specific to particular personality trait, while BoW produced 13.514 features, with each word in the dataset being treated as a feature, and its frequency in each tweet counted.

3.3 Model Training and Evaluation

In this subsection, evaluation will be done, which follows the planned modeling steps. The aim of the evaluation is to investigate the performance of the SVM and Naive Bayes algorithm models by using predefined test data ratios. The results are evaluated with a complete evaluation matrix containing parameters such as accuracy, F1-score, precision, and recall. These metrics describe the efficacy of the models in various test scenarios and datasets.

The evaluation conclusions are done by the calculation of weighted averages of metrics of F1-score, Recall, Accuracy, and Precision. This allows for even deeper insight into the performance of the models at different magnitudes of testing conditions and data sets. In this approach, weighted averages allow more effective comparison of overall performance results of the models by taking the importance of each metric into consideration. This will also enable a more balanced view of strengths and weaknesses for the models, since it might well be that one metric is higher at the expense of another. It will also be useful for indicating areas in which both models might be improved-for example, potential trade-offs between recall and precision are highly relevant in practical applications.

This is shown in Tables 5, 6, and 7, where a comparison of performance by different Support Vector Machine models with respect to BoW, BoW + LIWC, TF-IDF, and TF-IDF + LIWC feature configurations for C = 0.1, 1, 10 and kernel variant (Linear, Poly, RBF, Sigmoid) is drawn. The differentials in the results of accuracy, precision, recall, and F1-score show the different impacts that different features and model parameters have on the performance of classification. This analysis has provided a fair amount of insight as to which of these configurations does the best, taking a look at the two configurations put forward in demonstrating feature selection and tuning parameters for optimal results on text classification tasks.

Table 5. Baseline SVM Model Result

Model	Data Configuration	Parameters		Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
		C	Kernel				
SVM	BoW	0.1	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	38.2979	32.8131	38.2979	23.6651
			RBF	38.2979	14.6673	38.2979	21.2111
			Sigmoid	38.2979	14.6673	38.2979	21.2111
		1	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	38.2979	38.9906	38.2979	26.5856
			RBF	38.2979	29.0321	38.2979	32.2499
			Sigmoid	40.4255	31.0875	40.4255	34.8936
		10	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	36.1702	24.5687	36.1702	26.677
			RBF	38.2979	28.7155	38.2979	32.5664
			Sigmoid	40.4255	42.5621	40.4255	41.1461
	BoW + LIWC	0.1	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	38.2979	32.8131	38.2979	23.6651
			RBF	38.2979	14.6673	38.2979	21.2111
			Sigmoid	38.2979	14.6673	38.2979	21.2111
		1	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	38.2979	38.9906	38.2979	26.5856
			RBF	36.1702	27.3661	36.1702	30.6198
			Sigmoid	42.5532	33.4752	42.5532	36.5691
		10	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	36.1702	24.5687	36.1702	26.677
			RBF	40.4255	30.1354	40.4255	34.458
			Sigmoid	48.9362	46.7288	48.9362	47.1222
	TF-IDF	0.1	Linear	38.2979	14.6673	38.2979	21.2111
			Poly	38.2979	14.6673	38.2979	21.2111
			RBF	38.2979	14.6673	38.2979	21.2111
			Sigmoid	38.2979	14.6673	38.2979	21.2111
		1	Linear	55.3191	42.373	55.3191	46.9322
			Poly	42.5532	31.7222	42.5532	36.3259

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

			RBF	48.9362	41.2467	48.9362	39.9462
			Sigmoid	53.1915	41.8209	53.1915	45.0321
		10	Linear	46.8085	36.4225	46.8085	40.9671
			Poly	40.4255	30.3250	40.4255	34.4993
			RBF	51.0638	38.8298	51.0638	44.1119
			Sigmoid	38.2979	33.8723	38.2979	35.1971
	TF-IDF + LIWC	0.1	Linear	65.9573	60.778	65.9573	62.7153
			Poly	46.8084	52.2019	46.8084	36.3812
			RBF	44.6809	51.8375	44.6809	33.0884
			Sigmoid	38.2979	14.6673	38.2979	21.2111
		1	Linear	76.5957	77.6832	76.5957	76.5642
			Poly	53.1915	53.4043	53.1915	44.8704
			RBF	74.4681	57.9318	74.4681	64.4713
			Sigmoid	46.8084	36.0460	46.8084	39.5319
		10	Linear	72.3404	69.7400	72.3404	70.0113
			Poly	61.7021	51.773	61.7021	54.9201
			RBF	72.3404	65.8502	72.3404	67.8573
			Sigmoid	38.2979	28.6524	38.2979	32.6624

Table 5 gives the results of the baseline model. The best accuracy, 76.60%, was achieved by using TF-IDF + LIWC at C=1 with a linear kernel. Thus, this configuration had high precision at 77.68% and recall at 76.60%, which gave evidence of the efficiency of being able to catch most of the positives and negatives. While the combination of TF-IDF and LIWC features together, coupled with a linear kernel, forms a pretty well-balanced and robust model, it really unleashes the potential of the proposed features for text classification tasks. The high score for all metrics demonstrates that this feature set will not only improve accuracy but also ensure the model has a reliable recall rate to extend its application in real-world scenarios where balanced performance is important.

Table 6. Oversampling SVM Model Result

Model	**Data Configuration**	**Parameters**		**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
		C	Kernel				
SVM	BoW	0.1	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	36.1702	32.3017	36.1702	22.8042
			RBF	14.8936	29.2336	14.8936	17.7242
			Sigmoid	36.1702	40.6547	36.1702	31.7745
		1	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	38.2979	51.7566	38.2979	26.9504
			RBF	44.6809	42.1947	44.6809	35.9878
			Sigmoid	42.5532	39.2494	42.5532	37.9927
		10	Linear	34.0426	31.0311	34.0426	31.6207
			Poly	42.5532	33.7515	42.5532	37.4247
			RBF	38.2979	28.7155	38.2979	32.5664
			Sigmoid	42.5532	39.4998	42.5532	40.4597
	BoW + LIWC	0.1	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	36.1702	32.3017	36.1702	22.8042
			RBF	14.8936	28.4195	14.8936	17.4698
			Sigmoid	36.1702	45.7811	36.1702	31.7697
		1	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	38.2979	51.7566	38.2979	26.9504
			RBF	44.6809	38.8395	44.6809	37.9649
			Sigmoid	40.4255	35.5839	40.4255	35.0398
		10	Linear	34.0426	30.0532	34.0426	31.5242
			Poly	40.4255	31.9149	40.4255	35.3191
			RBF	40.4255	30.1354	40.4255	34.458
			Sigmoid	42.5532	46.1211	42.5532	42.5981
	TF-IDF	0.1	Linear	38.2979	31.1348	38.2979	33.1783
			Poly	40.4255	39.5068	40.4255	27.4552
			RBF	40.4255	42.5025	40.4255	30.0626
			Sigmoid	55.3191	48.7704	55.3191	50.2337
		1	Linear	53.1915	40.5154	53.1915	45.878
			Poly	42.5532	31.7222	42.5532	36.3259
			RBF	57.4467	44.5340	57.4467	49.4794
			Sigmoid	59.5745	51.6717	59.5745	53.2377

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

		10	Linear	46.8084	36.4225	46.8084	40.9671
			Poly	40.4255	30.3250	40.4255	34.4993
			RBF	51.0638	38.8298	51.0638	44.1119
			Sigmoid	38.2979	33.8723	38.2979	35.1971
	TF-IDF + LIWC	0.1	Linear	70.2128	67.9184	70.2128	68.5549
			Poly	44.6809	45.461	44.6809	36.856
			RBF	40.4255	46.4007	40.4255	38.8722
			Sigmoid	27.6596	48.5419	27.6596	29.9721
		1	Linear	72.3404	69.7400	72.3404	70.0113
			Poly	51.0638	53.6778	51.0638	46.2722
			RBF	61.7021	54.2472	61.7021	57.7286
			Sigmoid	21.2766	32.6389	21.2766	24.2531
		10	Linear	72.3404	69.7400	72.3404	70.0113
			Poly	61.7021	52.4773	61.7021	55.4633
			RBF	72.3404	65.8502	72.3404	67.8573
			Sigmoid	19.1488	22.8747	19.1488	19.7056

The results in Table 6, corresponding to the oversampling model, give TF-IDF + LIWC as the best result of the feature sets in both conditions, with a slightly diminished performance from the baseline. This had an accuracy of 72.34%, but its precision was much lower, going as low as 69.74%, and a recall at 72.34% also. However, the overall F1-score of 70.01% did not change from the baseline, which implies that oversampling did not change much in performance balance but did add more to train from. Stability in this case would suggest the robustness of the TF-IDF + LIWC configuration to changes in the amount of data. This increase in data volume due to oversampling allowed the model to retain its capacity for effective classification of personality traits, despite the slight reduction in precision. This also means that the TF-IDF + LIWC feature set is inherently strong enough to perform consistently across different balancing techniques. Consistency in performance strengthens this belief: feature selection must be done to get stable and reliable results from a model.

Table 7. Undersampling SVM Model Result

Model	**Data Configuration**	**Parameters**		**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
		C	**Kernel**				
SVM	BoW	0.1	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	19.1488	41.3239	19.1488	9.1699
			RBF	36.1702	21.0775	36.1702	26.6336
			Sigmoid	42.5532	56.1251	42.5532	37.2655
		1	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	17.0213	41.0058	17.0213	8.6139
			RBF	36.1702	21.0775	36.1702	26.6336
			Sigmoid	42.5532	56.1251	42.55320	37.2655
		10	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	36.1702	51.4507	36.1702	23.0059
			RBF	40.4255	39.3251	40.4255	35.9751
			Sigmoid	36.1702	42.5618	36.1702	29.1836
	BoW + LIWC	0.1	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	19.1488	41.3239	19.1488	9.1699
			RBF	36.1702	21.0775	36.1702	26.6336
			Sigmoid	42.5532	56.1251	42.5532	37.2655
		1	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	17.0213	41.0058	17.0213	8.6139
			RBF	36.1702	21.0775	36.1702	26.6336
			Sigmoid	42.5532	56.1251	42.5532	37.2655
		10	Linear	40.4255	61.7954	40.4255	31.8569
			Poly	36.1702	51.4507	36.1702	23.0059
			RBF	38.2979	38.4974	38.2979	34.648
			Sigmoid	34.0426	42.3074	34.0426	28.3768
	TF-IDF	0.1	Linear	19.1488	41.6013	19.1488	22.7722
			Poly	23.4043	32.1277	23.4043	25.77
			RBF	21.2766	46.8036	21.2766	25.833
			Sigmoid	19.1488	41.6013	19.1488	22.7722
		1	Linear	21.2766	44.22	21.2766	25.3511
			Poly	21.2766	28.1272	21.2766	24.2096
			RBF	21.2766	46.8036	21.2766	25.8332

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

		Sigmoid	19.1488	41.6013	19.1488	22.7722
	10	Linear	21.2766	25.4110	21.2766	22.615
		Poly	21.2766	28.1272	21.2766	24.2096
		RBF	21.2766	25.4110	21.2766	22.615
		Sigmoid	23.4043	26.1456	23.4043	24.1967
TF-IDF + LIWC	0.1	Linear	29.7872	54.3760	29.7872	33.2057
		Poly	42.5532	51.6717	42.5532	32.5532
		RBF	31.9149	48.4347	31.9149	36.7592
		Sigmoid	25.5319	33.6170	25.5319	24.8227
	1	Linear	27.6596	55.5916	27.6596	32.0894
		Poly	44.6809	51.6717	44.6809	35.4813
		RBF	29.7872	48.2799	29.7872	35.2235
		Sigmoid	29.7872	45.0266	29.7872	31.7462
	10	Linear	27.6596	55.5916	27.6596	32.0894
		Poly	25.5319	39.2253	25.5319	21.5112
		RBF	23.4043	48.4696	23.4043	29.6253
		Sigmoid	21.2766	26.0502	21.2766	21.4534

On the other hand, the performances of the undersampling model, which are presented in Table 7, showed a significant fall in all settings. The top accuracy was 40.43% for TF-IDF + LIWC at C = 0.1 with a linear kernel, while recall and F1-score were considerably lower at 40.43% and 31.86%, respectively. These results indicate that the undersampling-as a strategy of reducing the dataset size by removing instances from the majority class-was detrimental to model performance. Lower recall and F1-score would hint at the model's positive instance recognition ability with the smaller and less balanced training data being compromised.

In the Naive Bayes analysis, as represented in Tables 8, 9, and 10, the performance of the model for different configurations, such as BoW, BoW + LIWC, TF-IDF, and TF-IDF + LIWC, and different values of the variance smoothing parameter, also shows great variation in accuracy, precision, recall, and F1-score. The TF-IDF + LIWC feature set performed better than the rest of the configurations, especially for the variance smoothing value 1xE-2, with an accuracy of 59.57%. This indicates that these features capture the meaningful pattern in the data. The performance improvement accentuates the fine-tuning of hyperparameters toward the optimal model performance. However, the Naive Bayes model, though performing well, did not perform as well as the SVM model and thus required careful parameter selection.

Table 8. Baseline Naive Bayes Model Result

Model	**Data Configuration**	**Parameters**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
		Var_Smoothing				
Naive Bayes	BoW	1xE-0	34.0426	13.1528	34.0426	18.9745
		1xE-1	31.9149	18.6578	31.9149	21.1799
		1xE-2	10.6383	26.2139	10.6383	8.9509
		1xE-3	14.8936	21.5426	14.8936	13.5765
		1xE-4	21.2766	30.9016	21.2766	22.3677
		1xE-5	27.6596	23.6322	27.6596	25.4877
		1xE-6	38.2979	29.9719	38.2979	32.3780
		1xE-7	38.2979	30.2377	38.2979	31.339
		1xE-8	38.2979	30.2377	38.2979	31.339
		1xE-9	38.2979	30.2377	38.2979	31.339
	BoW + LIWC	1xE-0	34.0426	13.1528	34.0426	18.9745
		1xE-1	31.9149	18.6578	31.9149	21.1799
		1xE-2	10.6383	26.2139	10.6383	8.9509
		1xE-3	14.8936	21.5426	14.8936	13.5765
		1xE-4	21.2766	30.9016	21.2766	22.3677
		1xE-5	27.6596	24.1081	27.6596	25.7592
		1xE-6	38.2979	29.9719	38.2979	32.3780
		1xE-7	38.2979	30.2377	38.2979	31.339
		1xE-8	38.2979	30.2377	38.2979	31.339
		1xE-9	38.2979	30.2377	38.2979	31.339
	TF-IDF	1xE-0	42.5532	27.5076	42.5532	32.0567
		1xE-1	4.2553	19.208	4.2553	3.9448
		1xE-2	17.0213	48.577	17.0213	17.4768
		1xE-3	29.7872	30.9878	29.7872	30.0439
		1xE-4	31.9149	25.2455	31.9149	27.9175
		1xE-5	38.2979	29.1793	38.2979	32.7475

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

		1xE-6	40.4255	33.8548	40.4255	34.0183
		1xE-7	40.4255	33.8548	40.4255	34.0183
		1xE-8	40.4255	33.8548	40.4255	34.0183
		1xE-9	40.4255	33.8548	40.4255	34.0183
	TF-IDF + LIWC	1xE-0	25.5319	44.2848	25.5319	26.5029
		1xE-1	48.9362	61.5436	48.9362	52.628
		1xE-2	59.5745	68.6407	59.5745	60.9229
		1xE-3	27.6596	60.7675	27.6596	36.3808
		1xE-4	29.7872	55.5943	29.7872	31.1469
		1xE-5	38.2979	36.8794	38.2979	37.482
		1xE-6	34.0426	26.0637	34.0426	29.1941
		1xE-7	40.4255	32.2525	40.4255	34.383
		1xE-8	40.4255	33.8548	40.4255	34.0183
		1xE-9	40.4255	33.8548	40.4255	34.0183

At baseline, the best performance of 42.55% in Table 8 is achieved when the TF-IDF configuration is at Var_Smoothing = 1xE-0 with precision at 27.51%, recall at 42.55%, and F1-score at 32.06%. Results improved with the decrease in the Var_Smoothing value, while the best results for TF-IDF + LIWC were achieved at Var_Smoothing = 1xE-2, thus yielding an accuracy of 59.57%, a precision of 68.64%, a recall of 59.57%, and an F1-score of 60.92%. This improvement signifies that with the given data, a low value for the variance smoothing helps the model grasp minute variations, more so when the feature set has been enriched. The increased precision and recall show that the inclusion of LIWC features with TF-IDF increases the model's performance in classifying personality traits. These results point out the importance of feature selection and hyperparameter tuning for optimal performance.

Table 9. Oversampling Naive Bayes Model Result

Model	Data Configuration	Parameters Var_Smoothing	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Naive Bayes	BoW	1xE-0	36.1702	20.1418	36.1702	23.4009
		1xE-1	8.5106	8.956	8.5106	4.2368
		1xE-2	8.5106	25.1417	8.5106	7.5183
		1xE-3	14.8936	22.435	14.8936	14.0468
		1xE-4	21.2766	28.7814	21.2766	22.4739
		1xE-5	29.7872	24.924	29.7872	27.1341
		1xE-6	36.1702	27.6152	36.1702	29.8929
		1xE-7	38.2979	30.2377	38.2979	31.339
		1xE-8	38.2979	30.2377	38.2979	31.339
		1xE-9	38.2979	30.2377	38.2979	31.339
	BoW + LIWC	1xE-0	36.1702	20.1418	36.1702	23.4009
		1xE-1	8.5106	8.956	8.5106	4.2368
		1xE-2	8.5106	25.1417	8.5106	7.5183
		1xE-3	14.8936	23.6102	14.8936	14.6612
		1xE-4	23.4043	32.5655	23.4043	24.5674
		1xE-5	29.7872	24.924	29.7872	27.1341
		1xE-6	36.1702	27.6152	36.1702	29.8929
		1xE-7	38.2979	30.2377	38.2979	31.339
		1xE-8	38.2979	30.2377	38.2979	31.339
		1xE-9	38.2979	30.2377	38.2979	31.339
	TF-IDF	1xE-0	10.6383	41.1922	10.6383	13.4858
		1xE-1	6.383	25.5851	6.383	7.3986
		1xE-2	21.2766	68.1738	21.2766	22.4644
		1xE-3	34.0426	33.7766	34.0426	33.6549
		1xE-4	29.7872	23.8298	29.7872	26.3236
		1xE-5	38.2979	29.1793	38.2979	32.7475
		1xE-6	40.4255	33.8548	40.4255	34.0183
		1xE-7	40.4255	33.8548	40.4255	34.0183
		1xE-8	40.4255	33.8548	40.4255	34.0183
		1xE-9	40.4255	33.8548	40.4255	34.0183
	TF-IDF + LIWC	1xE-0	19.1488	39.1	19.1488	17.6249
		1xE-1	23.4043	46.3415	23.4043	25.1556
		1xE-2	55.3191	65.1807	55.3191	55.6231
		1xE-3	57.4467	69.2799	57.4467	59.7985

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

	1xE-4	23.4043	75.818	23.4043	31.5317
	1xE-5	34.0426	43.921	34.0426	35.9844
	1xE-6	34.0426	28.7233	34.0426	31.0029
	1xE-7	38.2979	29.7872	38.2979	33.1915
	1xE-8	40.4255	32.6886	40.4255	33.9442
	1xE-9	40.4255	33.8548	40.4255	34.0183

The oversampling results also showed a similar trend for TF-IDF + LIWC outperforming all the other configurations, as can be seen in Table 9. For the best results of accuracy of 57.45%, the precision is 69.28%, recall is 57.45%, and the F1-score is 59.80% - were achieved when Var_Smoothing = 1xE-3. It has to be taken into consideration that oversampling has not worked consistently throughout most configurations. This also means that some configurations, like BoW and TF-IDF with higher smoothing values, showed slight decreases in accuracy and precision, proof that oversampling does not always imply better results.

Table 10. Undersampling Naive Bayes Model Result

Model	Data Configuration	Parameters Var_Smoothing	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Naive Bayes	BoW	1xE-0	36.1702	51.4507	36.1702	23.0059
		1xE-1	40.4255	69.0983	40.4255	31.0598
		1xE-2	42.5532	66.7484	42.5532	35.7792
		1xE-3	38.2979	59.6624	38.2979	35.6334
		1xE-4	27.6596	50.6664	27.6596	33.7667
		1xE-5	17.0213	43.9009	17.0213	20.3466
		1xE-6	17.0213	43.4172	17.0213	20.2634
		1xE-7	17.0213	43.4172	17.0213	20.2634
		1xE-8	14.8936	40.3979	14.8936	17.1539
		1xE-9	14.8936	40.3979	14.8936	17.1539
	BoW + LIWC	1xE-0	36.1702	51.4507	36.1702	23.0059
		1xE-1	40.4255	69.0983	40.4255	31.0598
		1xE-2	42.5532	66.7484	42.5532	35.7792
		1xE-3	42.5532	63.3232	42.5532	41.847
		1xE-4	27.6596	50.6825	27.6596	33.7939
		1xE-5	17.0213	43.9009	17.0213	20.3466
		1xE-6	17.0213	43.4172	17.0213	20.2634
		1xE-7	17.0213	43.4172	17.0213	20.2634
		1xE-8	14.8936	40.3979	14.8936	17.1539
		1xE-9	14.8936	40.3979	14.8936	17.1539
	TF-IDF	1xE-0	27.6596	16.8867	27.6596	20.7885
		1xE-1	25.5319	16.3004	25.5319	19.5634
		1xE-2	25.5319	35.1062	25.5319	25.9676
		1xE-3	14.8936	22.1188	14.8936	15.6705
		1xE-4	14.8936	19.2819	14.8936	14.1947
		1xE-5	12.766	15.1361	12.766	11.5668
		1xE-6	8.5106	8.595	8.5106	6.458
		1xE-7	8.5106	8.595	8.5106	6.458
		1xE-8	10.6383	18.5494	10.6383	10.0304
		1xE-9	10.6383	18.5494	10.6383	10.0304
	TF-IDF + LIWC	1xE-0	38.2979	50.8046	38.2979	41.2802
		1xE-1	36.1702	58.463	36.1702	41.9266
		1xE-2	38.2979	74.4076	38.2979	46.0809
		1xE-3	40.4255	74.1675	40.4255	50.135
		1xE-4	31.9149	66.8262	31.9149	40.0453
		1xE-5	21.2766	57.0522	21.2766	27.0212
		1xE-6	12.766	20.3079	12.766	13.4092
		1xE-7	12.766	20.7747	12.766	12.1908
		1xE-8	8.5106	10.7198	8.5106	6.6358
		1xE-9	8.5106	10.7198	8.5106	6.6358

Table 10 illustrates the results of undersampling, which have an obvious decline in performance compared to the baseline and oversampling models. The best result using TF-IDF is at Var_Smoothing = 1xE-2, with the highest accuracy of 42.55%, but the precision and recall were lower, hence reducing the F1-scores. This means that the undersampling technique impeded the model from effectively classifying positive instances. The further decrease in

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

recall and F1-score shows that undersampling might result in a worse model because it will lose some valuable data, which might become helpful for the model.

All these together indicate that TF-IDF+LIWC has indeed the best performance among all the representations for this task, at least concerning the metrics of accuracy, precision, recall, and F1-score, with optimal performance generally obtained with small values of Var_Smoothing for Naive Bayes and linear kernel for the SVM classifier. In Naive Bayes, the best results were achieved for a variance smoothing value of 1xE-2, which means that a medium level of smoothing provides a more effective trade-off between the capturing of subtle patterns in the data and preventing overfitting. The best performing results obtained by using the linear kernel with the SVM model were TF-IDF + LIWC, ranking best on all datasets. Generally, oversampling increased recall and precision, although this was most evident using the TF-IDF + LIWC feature set. In some cases, oversampling led to slight reductions in accuracy or precision, suggesting that the method may not always be beneficial, especially when used with configurations that already perform well. Meanwhile, the undersampling technique had a negative impact on model performance, especially for recall and F1-score, since it caused a loss of important data. That again underlines the importance of the choice of data balancing technique, since this choice can significantly affect the model's ability to classify personality traits effectively. It also indicates that careful experimentation with both feature selection and data preprocessing techniques is necessary to optimize overall performance.

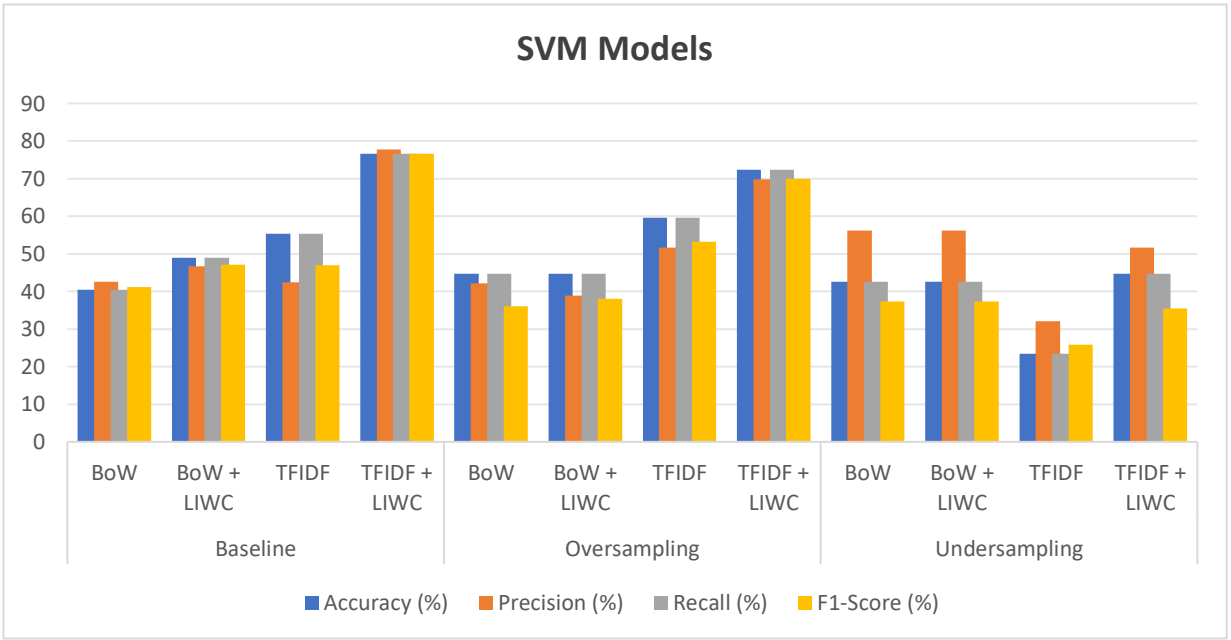

Figure 2. SVM Model Summary

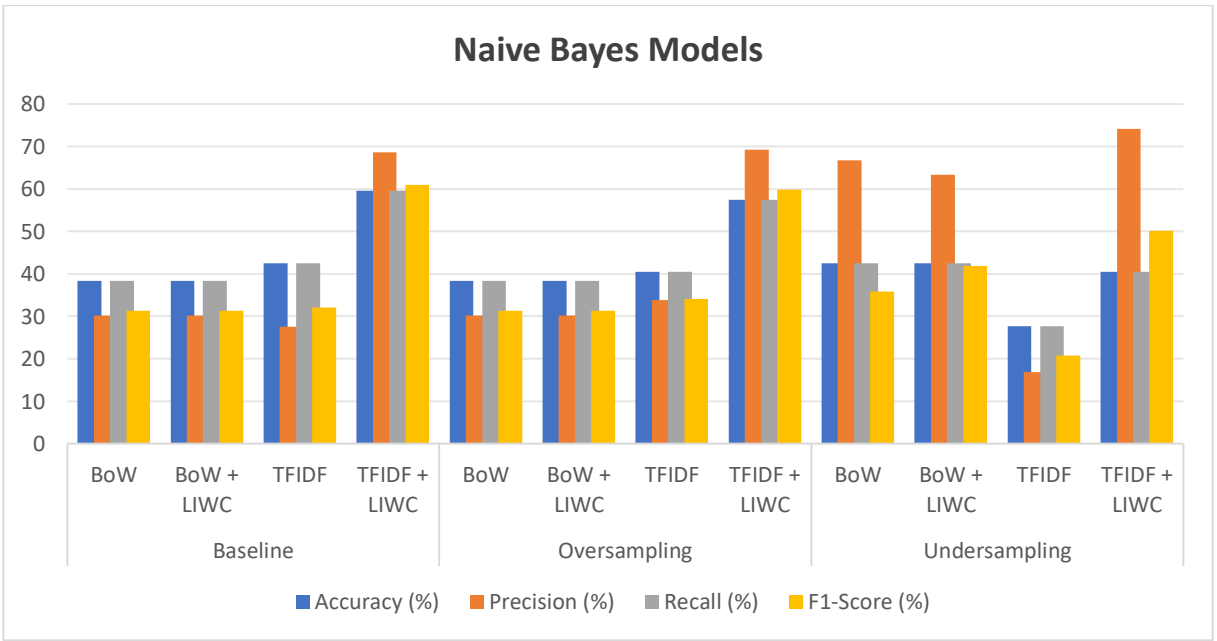

Figure 3. Naive Bayes Model Summary

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

The results are shown in Figures 2 and 3, which represent the performance of the SVM and Naive Bayes models for different configurations and balancing scenarios. In the case of the SVM model, represented in Figure 2, it is clear that the configuration TF-IDF + LIWC outperforms all the others in all metrics for the highest accuracy, precision, recall, and F1-score, especially in the baseline and oversampling scenarios. With this feature set, the linear kernel is remarkably successful, and this clearly gives it an edge in the classification task. By comparison, Figure 3 shows that the Naive Bayes model also improves with the TF-IDF + LIWC feature set, especially in the undersampling configuration, where precision and recall are considerably improved. However, the overall accuracy and F1-score of the Naive Bayes model are below those of the SVM model. These results point to the superior performance of the TF-IDF + LIWC feature combination, especially when combined with the SVM model across all scenarios, thus underlining the importance of feature expansion and model selection for personality classification from social media data.

3.4 Discussion

The results of this study, as shown in Tables 5 to 10, give a very important idea about the relative performance of the Support Vector Machine and Naive Bayes models using different feature sets and data balancing techniques. Both models showed that combining TF-IDF and LIWC features gave the best performance time after time, demonstrating the value of integrating both traditional text features and linguistic insights. This set of hybrid features enables accuracy and interpretability, which enables state-of-the-art performance on tasks of interest.

The best performance for the SVM model was achieved at the configuration of TF-IDF + LIWC, where the value of C was 1, using a linear kernel, giving an accuracy of 76.60%, precision of 77.68%, and recall of 76.60%, as shown in Table 5. This configuration thus had a well-balanced performance in terms of precision and recall, indicating that both the positive and negative instances were well captured by the model. While the oversampling technique slightly lowered the performance of the model, the F1-score was consistent at 70.01%, indicating that the set of features was still solid even when the dataset had been modified. On the contrary, the undersampling model performed significantly worse, with a peak accuracy of 40.43% at C=0.1 and with a linear kernel; this really underlined the negative effect of undersampling on the effectiveness of SVM.

For the Naive Bayes model, the best performance was achieved for the TF-IDF + LIWC configuration with a variance smoothing value of 1xE-2, with an accuracy of 59.57%, precision of 68.64%, recall of 59.57%, and F1-score of 60.92%, as presented in Table 8. That means that, similar to the SVM model, a combination of TF-IDF and LIWC features was the strongest feature set for Naive Bayes. In the case of an oversampling model, the best performance, with a smoothing parameter equal to 1xE-3, reached an accuracy of 57.45% and F1-score of 59.80%, although it did not show a consistent improvement over the baseline. It follows that the undersampling approach negatively affected the performance by yielding lower accuracy and recall; hence, the undersampling technique resulted in a less effective model that could not catch the fine nuances of the data.

Throughout both models, the TF-IDF + LIWC feature set performed higher in accuracy, precision, recall, and F1-score than other configurations. This emphasizes the importance of integrating statistical text features with linguistic and psychological indicators, something which is highly required for tasks such as sentiment analysis and emotion detection. The results also show that while oversampling had some stabilizing effect on recall and precision, undersampling generally had a negative effect on both models. This loss was especially prominent in terms of recall, which shows that the majority class information is important to retain to avoid losing classification effectiveness.

In summary, this study highly underlines the feature selection, model type, and data balancing technique. For instance, the TF-IDF + LIWC feature set gave the best results with a slight difference from NB, especially in the baseline and oversampling models using the SVM algorithm. On comparing the two balancing strategies in this study, it has also been found that the class balance strategy of oversampling tends to maintain stability whereas undersampling can seriously decrease model performance, especially on Recall and F1-score. Future work may be related to testing alternative strategies for balancing the data or incorporating more feature sets that would enhance the capability of the model to handle imbalanced data.

4. CONCLUSION

This work proved that the proposed combination of TF-IDF and LIWC features improves the performance of both the SVM and Naive Bayes models in text classification tasks. Among the tested models, the SVM with a configuration of TF-IDF + LIWC had the highest accuracy of 76.60% in the baseline model and was also very consistent in both the baseline and oversampling scenarios. This model also manifested balanced precision, recall, and F1-score, underlining its efficacy in handling the text data of complex features. Simultaneously, the performance of Naive Bayes was best at TF-IDF + LIWC when Var_Smoothing = 1xE-2, while accuracy resulted in 59.57%, and it showed lower precision as compared to the SVM model. This results in a slight oversampling approach that raises the recall and precision, mostly when combined with the TF-IDF + LIWC configuration. However, it is not outperforming the baseline consistently. The under-sampling technique performed very badly for the models, mostly impacting the recall and F1-score of the overall performances. In general, this study has proved that the TF-IDF + LIWC feature set, especially in the SVM model with a linear kernel, has consistently provided the best results, showing that feature configuration and

Building of Informatics, Technology and Science (BITS)
Volume 6, No 3, Desember 2024 Page: 1717−1731
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v6i3.6411

methods of sampling bear a critical role in enhancing the accuracy of classification and effectiveness of models. These findings have underlined the careful selection of both feature representations and data balancing techniques in order to optimize model performance in text classification tasks. Most important is a reminder that model configuration and sampling strategy must be optimized for the particular classification challenge at hand, and perhaps through research that refines these further, alternative approaches may arise, in due time, to further improve performance in text classification.

REFERENCES

[1] H. Perera and L. Costa, "Personality Classification of text through Machine learning and Deep learning: A Review," Int. J. Res. Adv. Comput. Sci. Eng., vol. 9, pp. 6–12, 2023, doi: 10.36227/techrxiv.22337746.
[2] S. R. M. and P. M. K. P. S. Dandannavar, "Social Media Text - A Source for Personality Prediction," in Proc. 2018 Int. Conf. Comput. Techn. Electron. Mech. Syst. (CTEMS), IEEE, Jul. 2019.
[3] M. Villeda and R. McCamey, "Use of Social Networking Sites for Recruiting and Selecting in the Hiring Process," Int. Bus. Res., vol. 12, no. 3, p. 66, Feb. 2019, doi: 10.5539/ibr.v12n3p66.
[4] J. Serrano-Guerrero, B. Alshouha, M. Bani-Doumi, F. Chiclana, F. P. Romero, and J. A. Olivas, "Combining machine learning algorithms for personality trait prediction," Egypt. Inform. J., vol. 25, Mar. 2024, doi: 10.1016/j.eij.2024.100439.
[5] F. Mairesse, M. A. Walker, M. R. Mehl, and R. K. Moore, "Using linguistic cues for the automatic recognition of personality in conversation and text," J. Artif. Intell. Res., vol. 30, pp. 457–500, 2007, doi: 10.1613/jair.2349.
[6] E. Ronando, M. Yasa, and E. Indasyah, "Sistem Prediksi Kepribadian Manusia Berdasarkan Status Media Sosial Menggunakan Support Vector Machine," Konvergensi, vol. 17, no. 1, pp. 13–22, Aug. 2021, doi: 10.30996/konv.v17i1.5164..
[7] R. Yunita and C. Sitasi, "Aktivitas Pengungkapan Diri Remaja Putri Melalui Sosial Media Twitter," Jurnal Komunikasi, vol. 10, no. 1, pp. 26–32, 2019. [Online]. Available: http://ejournal.bsi.ac.id/ejurnal/index.php/jkom.
[8] Y. B. N. D. Artissa, I. Asror, and S. A. Faraby, "Personality Classification based on Facebook status text using Multinomial Naïve Bayes method," in Proc. J. Phys. Conf. Ser., Institute of Physics Publishing, May 2019, doi: 10.1088/1742-6596/1192/1/012003.
[9] A. Nugroho and Y. Religia, "Analisis Optimasi Algoritma Klasifikasi Naive Bayes menggunakan Genetic Algorithm dan Bagging," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 5, no. 3, pp. 504–510, Jun. 2021, doi: 10.29207/resti.v5i3.3067.
[10] A. P. Natasuwarna, "Analisis Sentimen Keputusan Pemindahan Ibu Kota Negara Menggunakan Klasifikasi Naive Bayes," in Sensitif 2019, Makasar, Dec. 2018, pp. 47–53.
[11] G. J. & S. H. Sanhaji, "WFraud Alert Sebagai Prediksi Pesan Penipuan WhatsApp Menggunakan Naïve Bayes," Tekno Kompak, vol. 18, pp. 113–125, 2020.
[12] F. F. Irfani, "Analisis Sentimen Review Aplikasi Ruangguru Menggunakan Algoritma Support Vector Machine," JBMI (Jurnal Bisnis, Manajemen, dan Informatika), vol. 16, no. 3, pp. 258–266, Feb. 2020, doi: 10.26487/jbmi.v16i3.8607.
[13] E. Indrayuni, A. Nurhadi, and D. A. Kristiyanti, "Implementasi Algoritma Naive Bayes, Support Vector Machine, dan K-Nearest Neighbors untuk Analisa Sentimen Aplikasi Halodoc," Faktor Exacta, vol. 14, no. 2, p. 64, Aug. 2021, doi: 10.30998/faktorexacta.v14i2.9697.
[14] A. Fikriani, I. Asror, and Y. R. Murti, "Klasifikasi Kepribadian Berdasarkan Data Twitter dengan Menggunakan Metode Support Vector Machine," in e-Proceeding of Engineering, 2019, pp. 10436–10450.
[15] J. W. Iskandar and Y. Nataliani, "Perbandingan Naïve Bayes, SVM, dan k-NN untuk Analisis Sentimen Gadget Berbasis Aspek," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 5, no. 6, pp. 1120–1126, Dec. 2021, doi: 10.29207/resti.v5i6.3588.
[16] K. Shin, J. Han, and S. Kang, "MI-MOTE: Multiple imputation-based minority oversampling technique for imbalanced and incomplete data classification," Inf. Sci. (N Y), vol. 575, pp. 80–89, 2021.
[17] A. Smith and B. Johnson, "Efficient parameter tuning for support vector machines in large-scale datasets," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 8, pp. 2404–2415, 2019.
[18] J. Lee, H. Park, and S. Kim, "Enhanced support vector machines using adaptive kernel functions," Pattern Recognit. Lett., vol. 131, pp. 123–130, 2020.
[19] N. S. Fauziah and R. D. Dana, "Implementasi algoritma Naive Bayes dalam klasifikasi status kesejahteraan masyarakat Desa Gunungsari," Blend Sains Jurnal Teknik, vol. 1, no. 4, pp. 295–305, Mar. 2023, doi: 10.56211/blendsains.v1i4.234.
[20] A. Budiman, J. C. Young, and A. Suryadibrata, "Implementasi algoritma Naive Bayes untuk klasifikasi konten Twitter dengan indikasi depresi," Jurnal Informatika: Jurnal Pengembangan IT, vol. 6, no. 2, 2021.