IJSSR
International Journal of Social Service and Research

Vol. 6, No. 4, April 2026
e-ISSN: 2807-8691 | *p*-ISSN: 2807-839X

Sentiment Analysis of Distance Learning Using the K-Nearest Neighbor Method

Ni Wayan Devina Maharani*, Fitrianingsih
Universitas Gunadarma, Indonesia
Email: niwayandevinam@gmail.com*, fitrianingsih@staff.gunadarma.ac.id

ABSTRACT

During the pandemic, the Indonesian government issued a Distance Learning (PJJ) policy to reduce the spread of COVID-19. Many people expressed opinions about the pros and cons of the implementation of distance learning policies through social media, one of which is Twitter. These opinions can then be processed by conducting sentiment analysis. In this study, researcher will implement the K-Nearest Neighbor method to conduct sentiment analysis on Twitter regarding distance learning. The initial stage of the research is collecting tweets from Twitter as many as 1014 data. The next stage is labeling the dataset manually, which is then followed by the preprocessing stage which consists of data cleaning, case folding, tokenization, normalization, stopword removal and stemming. The dataset is further divided into two, namely train data and test data using an 8:2 ratio, where 80% is used as train data and 20% is used as test data. The K-Nearest Neighbor model is then built with several different hyperparameters. The KNN model evaluated using test data. The calculation of the accuracy value between the prediction sentiment and the actual sentiment of the test data is done using confusion matrix. The results of data classification using the K-Nearest Neighbor method with the most optimal hyperparameter resulted in an accuracy of 74.38%. The results of the study are expected to be able to classify positive and negative sentiment within sentences with the best accuracy so that the results of this study can help the government regarding distance learning policies during the pandemic.

Keywords: Sentiment Analysis, Twitter, Distance Learning, K-Nearest Neighbor

INTRODUCTION

The COVID-19 virus (Coronavirus Disease 2019) that first occurred in the city of Wuhan, China on December 30, 2019 has become a pandemic outbreak that has lasted for more than a year. The Indonesian government issued a Distance Learning (PJJ) policy as one of the efforts to reduce the spread of COVID-19. The Distance Learning Policy that has been implemented since March 16, 2020 has a significant impact on the education sector in Indonesia, especially on students. A survey conducted by UNICEF (United Nations Children's Fund) in June 2020 with 4000 students throughout Indonesia showed that 38% of respondents felt a lack of guidance from teachers and 35% experienced poor internet access when conducting Distance Learning (source: https://www.unicef.org/, 2021). Distance Learning also has a positive impact on technological and communication advancements, as all educational activities are carried out online (Adriani et al., 2007). This impact has made the Distance Learning policy bring positive and negative opinions from the public.

Many people expressed their pro and con opinions on the implementation of the distance learning policy through social media, one of which was Twitter. Twitter is a social media that is often used to share opinions with the use of hashtags that make it easier for users to search for certain topics. These opinions can then be processed by conducting sentiment analysis. The sentiment contained in an opinion can be positive or negative, so the opinion

can be classified based on the description of the opinion on the sentiment, whether the opinion tends to have a positive or negative sentiment aspect.

One method that can be used to classify sentiment in an opinion is the K-Nearest Neighbor (KNN) method. The K-Nearest Neighbor algorithm is a method used to classify objects based on learning data with the closest distance to the object (Liantoni, 2016). Research using the KNN method has been carried out by several previous researchers. The researcher (Liantoni, 2016) used the KNN method using Euclidian distance. The data used is leaf image data. Research by (Septian, Fachrudin, and Nugroho, 2019) used the KNN method to analyze sentiment using TF-IDF for word weighting and Confusion Matrix for accuracy calculation. The study (Khan, Kanwal, Alamri, and Mumtaz, 2020) used the KNN method that was optimized using hyperparameters. The results of using these hyperparameters are better than the ordinary KNN method. Researchers (Nurfarida, Indriati, and Perdana, 2018) conducted a study using the process of cleansing, case folding, filtering, tokenization, and stemming in data preprocessing to determine sentiment using the KNN method. The research conducted by (Rezwanul, Ali, and Rahman, 2017) compared the Support Vector Machine method with KNN to conduct sentiment analysis on data taken from Twitter social media and get better accuracy using KNN. The study (Sudira, Diar, and Ruldeviyani, 2019) also compared the KNN method with the Naïve Bayes method in sentiment analysis on the satisfaction level of users of digital payment services. The data used was taken from comments on Instagram social media. The accuracy obtained by the KNN method in the research is better than the Naïve Bayes method. The results of some of these studies show that the K-Nearest Neighbor method is suitable for use in conducting sentiment analysis.

In this study, the researcher will implement the K-Nearest Neighbor method to conduct sentiment analysis on Twitter social media regarding distance learning. This study uses hyperparameters to improve the performance of the model in conducting sentiment analysis. The hyperparameters used in this study are k-value, distance metric, and weights. The results of the study are expected to classify positive and negative sentences with the best accuracy so that the results of this study can help the government regarding distance learning policies during the pandemic.

METHODS

In this study, several stages were carried out to create a sentiment analysis system on the topic of Distance Learning. The method used in this study can be seen in Figure 3.1.

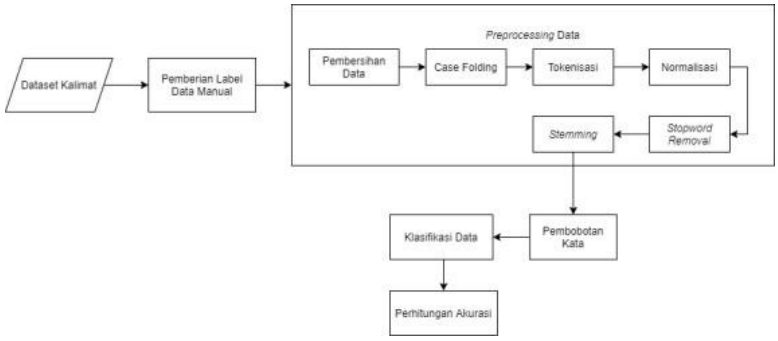

Figure 3.1. Research Methods

The initial stage carried out in this study was the stage of collecting data on tweet retrieval from social media Twitter using the *hashtag* "#pembelajaranjarakjauh" and the keyword "distance learning". The next stage is the manual labeling of the data, which is then followed by *the preprocessing* stage. The *preprocessing* stage consists of data cleaning, *case folding*, tokenization, normalization, *stopword removal* and *stemming*. After the *preprocessing* stage, word weighting is then carried out using TF-IDF. The data is then divided into two types, namely data *train* used in the *K-Nearest Neighbor model training* and test data used to evaluate the model that has been trained. The division uses an 8:2 ratio, where 80% is used as a data *train* and 20% is used as a test data. Then, the *K-Nearest Neighbor model* was formed with several *different hyperparameters*. The model that has been designed is then trained with data *train*. The output of this process will result in a *K-Nearest Neighbor* model that has been trained and can be used for data classification. The KNN model will be evaluated using a data *test* with *the Confusion Matrix*. The results of the *Confusion Matrix* are used for the calculation of the accuracy of the model. The last stage is to design an application in the form of *a website* for the implementation of the KNN model that has been evaluated.

Dataset of Sentences with *Hashtags*

The data collection stage was carried out using *Twitter Archiving Google Sheet* with the keyword Distance Learning and *hashtag* #pembelajaranjarakjauh. The steps in the data collection process are as follows.

1. The first step is to copy the spreadsheet to the Google Drive provided on the TAGS website (https://tags.hawksey.info/). Next, press the *get TAGS* button on the main page.
2. On the next page, two different versions of TAGS are given, namely version 6.0 and version 6.1. In this study, version 6.1 was used.
3. After selecting the TAGS version, a notification appears to ask for permission to copy the spreadsheet to *Google Drive*, then enter the keywords used and set the number of *tweets* to be saved
4. If the data filling stage is complete, then it is continued by giving *Google Drive* and Twitter access to the *spreadsheet*
5. If you have obtained the data, the data that has been collected can be seen on the *Archive sheet*. The results of this Twitter data collection were obtained by 1014 *tweets*.

Data Labeling

The label formation stage in this study is carried out manually, namely by classifying the data that has been collected into positive and negative categories without using a specific program. The manual method was chosen so that the labeling process on each data was more accurate even though it took quite a lot of time. Data that has been collected and is in the form of *a spreadsheet* is added to the column for the label which will later be filled with positive or negative labels according to the content of each data. Each of the labels used in this study will be explained as follows:

a. Positive label: in the form of text data that contains support or defense of the distance learning policy.
b. Negative label: in the form of text data that contains bad opinions, ridicule, sarcasm, or insults to distance learning policies.

The results of this stage produced 576 positive data and 438 negative data out of a total of 1014 data. An example of labeled text data can be seen in Table 1.

Table 1 Examples of Labeled Data

Tweet Text	Label
I'm an introvert, I'm willing to extend distance learning and not complain	Positive
Prolonged distance learning will result in learning loss in students. #ingatpesanibu #cucitangan #pakaimasker #jagajarak https://t.co/JhlxqXxwDv	Negatives
Conditions in rural areas are worse because distance learning does not work https://t.co/IxlbuzNLo9	Negatives

***Preprocessing* Data**

Preprocessing is done with the aim of transforming the raw data into the format needed for the sentiment analysis process. This *preprocessing* stage consists of several stages, namely data cleaning, *case folding*, tokenization, normalization, *stopword removal*, and *stemming*. Some of these processes will be explained as follows.

Data Cleansing

Tweet *data* obtained from Twitter often contains unnecessary components, such as *delimiters*, symbols, *emoticons*, or URLs, so the data needs to be cleaned up first. The steps at the data cleanup stage can be seen in Figure 1.

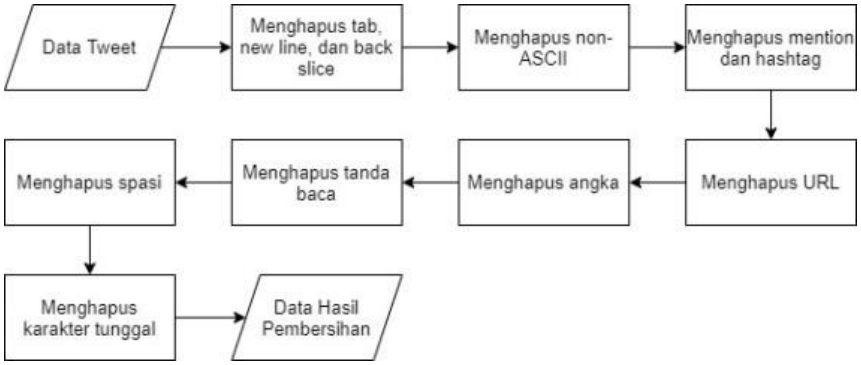

Figure 1 Data Cleanup Stage

The following is a comparison between the uncleaned data and the cleaned data can be seen in Table 2.

Table 2 Comparison of Data Before and After Data Cleansing

Before Data Cleansing	After Data Cleanup
I'm an introvert, I'm willing to extend distance learning and not complain	I'm an introvert, I'm willing to extend distance learning and not complain
Prolonged distance learning will result in learning loss in students. #ingatpesanibu #cucitangan #pakaimasker #jagajarak https://t.co/JhlxqXxwDv	Prolonged distance learning will result in learning loss in students
Conditions in rural areas are worse because distance learning does not work https://t.co/IxlbuzNLo9	Conditions in rural areas are worse because distance learning is not running

Based on the comparison in Table 2, it can be seen that the result of this data cleansing can remove *emoticons* in the first text, *hashtags* and URLs in the second text, and URLs in the third text.

Case Folding

The *case folding* stage is the stage to change the capital letters in the text to lowercase letters. The steps at the *case folding* stage can be seen in Figure 2.

Figure 2 Case *Folding Stage*

The following is a comparison between data that has not gone through the *case folding* stage and data that has gone through *the case folding* stage which can be seen in Table 3.

Table 3 Comparison of Data Before and After *Case Folding*

Before *Case Folding*	**After *Case Folding***
I'm an introvert, I'm willing to extend distance learning and not complain	I'm an introvert, I'm willing to extend distance learning and not complain
Prolonged distance learning will result in learning loss in students	prolonged distance learning will result in learning loss in students
Conditions in rural areas are worse because distance learning is not running	Conditions in rural areas are worse because distance learning does not run

Based on the comparison in Table 3, it can be seen that words such as "Learning" in the second text and the word "Condition" in the third text are changed to lowercase letters in the *case folding process*.

Tokenization

Tokenization is the process of separating each word that makes up a text. Text that has previously gone through *the case folding* stage will be divided or cut based on words into pieces that can be called tokens. In Python, the tokenization stage can be done with *the word_tokenize* function in *the nltk* library.*The Natural Language Toolkit* (NTLK) is a *library* that is commonly used to facilitate text processing activities. The steps on the tokenization process can be seen in Figure 3.9.

Figure 3 Tokenization Stage

Table 4 Comparison of Data Before and After Tokenization

Before Tokenization	After Tokenization
I'm an introvert, I'm willing to extend distance learning and not complain	['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']
prolonged distance learning will result in learning loss in students	['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']
Conditions in rural areas are worse because distance learning does not run	['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']

Data Normalization

Data normalization is used to convert text containing abbreviated words into the original word so that the word can be understood by the model. The steps at the normalization stage can be seen in Figure 4.

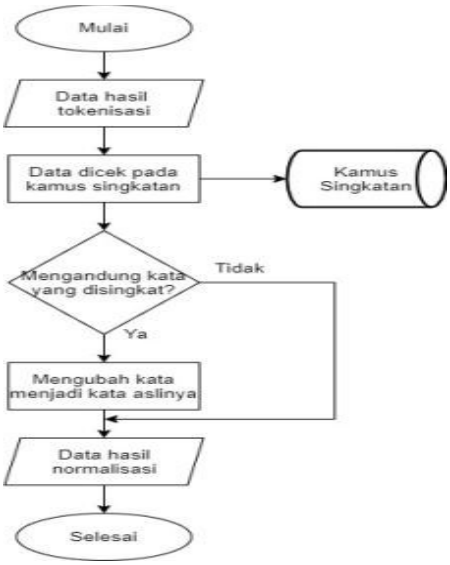

Figure 4 Data Normalization Stage

The following is a comparison between the unnormalized data and the normalized data which can be seen in Table 5.

Table 5 Comparison of Data Before and After Normalization

Before Normalization	After Normalization
['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']	['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']
['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']	['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']
['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']	['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']

Based on the comparison in Table 5, it can be seen that after going through the process of normalization, the third text that has an abbreviated word, namely "because" changes to the original word, namely "because".

K-Nearest Neighbor *Model Creation*

The next stage is the formation of a model using *the K-Nearest Neighbor* method. Before doing this step, the first thing that needs to be done is to do *hyperparameter tuning*. *Hyperparameter tuning* is carried out to determine the best *hyperparameters* that match the characteristics of the training data so that it can produce a model with good generalization capabilities. In this study, *the hyperparameter tuning* stage was carried out using *GridSearchCV*. *GridSearchCV* is a method for finding the best combination of *hyperparameter* values in a *grid*. *GridSearchCV* will divide the data according to the specified number of grids and train the model based on that data division. Then *GridSearchCV* will calculate the average accuracy of the training and select the parameters with the best accuracy.

In this study, the number of grids used in model training is 3 *grids*. The hyperparameters used in this study are the parameters of k-value, *weights*, and distance metrics. The best accuracy values obtained on *hyperparameter tuning* will be used in model training and testing. Here's a syntax snippet to define *hyperparameter* values and model formation:

```
knn = KNeighborsClassifier()
k_range = list(range(1,31))
weights = ['uniform', 'distance']
metric = ['euclidean','cosine']
param_grid = { 'n_neighbors' : k_range,
               'metric' : metric,
               'weights' : weights}
model = GridSearchCV(knn, param_grid, verbose = 1, cv=3, n_jobs =
-1,scoring='accuracy')
model.fit(X_train, y_train)

print(model.best_score_, model.score(X_test, y_test))
print(model.best_params_)
```

Description :

a. knn is a variable that stores *the K-Nearest Neighbor method.*
b. k_range is a k-value variable used in *the hyperparameter tuning* process.
c. Weights are variables that store the method of weighting neighbors for each neighbor in the *hyperparameter tuning process*.
d. Metric is a variable that stores the method of calculating the distance of each neighbor in the *hyperparameter tuning process*.
e. The model is used to host the model and the best parameter search results from *GridSearchCV*.
f. model.fit is used to perform the *process of fitting* the data *train* into the model.

Model Creation with K Value Parameters

The k-value in the K-Nearest Neighbor method indicates the number of closest neighbors needed to determine the classification of a sentence. The k-value parameter used at this stage consists of 30 values from 1 to 30. Other parameters are given default values used in the sklearn library, i.e. *Minkowski* for the distance metric parameter and *uniform* for weighting. The results of the model training with the k-value parameter can be seen in Table 6.

Table 6 Training Results with K Value Parameter

K value	Average *Training Accuracy* (%)	K value	Average *Training Accuracy* (%)	K value	Average *Training Accuracy* (%)
1	51,42%	11	67,57%	21	68,68%
2	44,39%	12	66,70%	22	69,67%
3	55,84%	13	68,31%	23	69,18%
4	52,52%	14	66,46%	24	68,56%
5	62,26%	15	68,06%	25	68,19%
6	59,79%	16	67,32%	26	68,31%
7	64,61%	17	68,43%	27	68,68%
8	62,64%	18	66,71%	28	69,30%
9	65,35%	19	67,94%	29	68,68%
10	64,73%	20	68,19%	30	69,42%

Based on Table 6 on the *record* given a red box, it can be seen that the highest *training* accuracy is achieved when using the value $k = 22$. Visualization of *training results using* the *k-value parameter* displayed in the form of a graph can be seen in Figure 5.

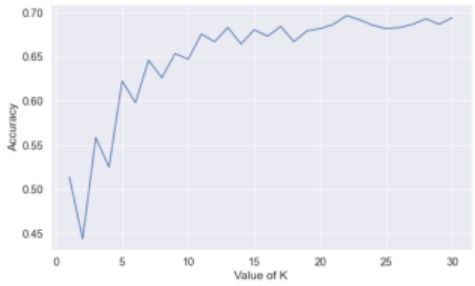

Figure 5 Training Graph with K Value Parameter

Figure 5 shows a graph of the accuracy results based on *the k-value* depicted with a blue line. The blue line moving up indicates that the greater the *value of k*, the greater the accuracy value. However, when the k-value has reached 22, the accuracy value begins to decrease, so it can be concluded that 22 is the most optimal k-value for data *training*.

Model Creation with Parameter *Weights*

The *parameter weights* in the K-Nearest Neighbor method is a method of weighting the neighbors of each sentence to be classified. The weights parameters used at this stage are *uniform weights* where the weight given to the neighboring sentence is equal to the same value and *distance weights* where the weight is given based on the distance from a sentence to its neighbor's sentence. The k-value parameter is given a value of 22 because it has the highest accuracy of the previous stage and the distance metric parameter is given the *default* value used on *the sklearn library, i.e.* Minkowski. The results of the model training with *the weights* parameter can be seen in Table 7.

Table 7 Training Results with Weights Parameters

Weight	**Average *Training Accuracy* (%)**
uniform	69,67%
distance	70,66%

Based on Table 7, it can be seen that the *training* accuracy is highest when using *distance weights*. The visualization of *training* results using the *weights* parameter displayed in the form of a graph can be seen in Figure 6.

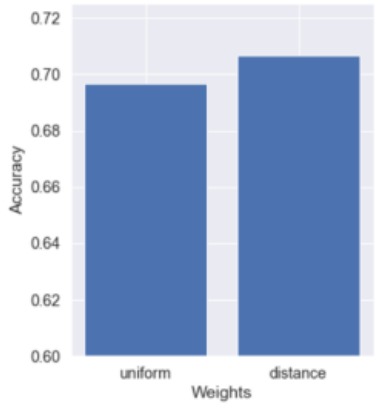

Figure 6 Training Chart with Weights Parameter

Figure 6 shows the graph of the accuracy results based on the 2 *weights* parameters used. The graph shows that the accuracy value of training using *distance weights* is better than the accuracy value using *uniform weights*.

Model Creation with Distance Metric Parameters

The distance metric in *K-Nearest Neighbor* is a method used to calculate the weight of the distance between sentences to be classified and their neighbors. The distance metrics used at this stage are *Euclidian Distance* and *Cosine Distance*. The parameter *value k* uses a value of 22 and *weights* use *distance weights* because it has the highest accuracy of the previous test. The results of the model training with the k-value parameter can be seen in Table 8.

Table 8 Training Results with Distance Metric Parameters

Distance Metrics	**Average *Training Accuracy* (%)**
Euclidian	70,66%
Cosine	68,80%

Based on Table 8, it can be seen that *the training* accuracy is highest when using *the Euclidian Distance* metric. Visualization of *training* results using distance metric parameters displayed in the form of a graph can be seen in Figure 7.

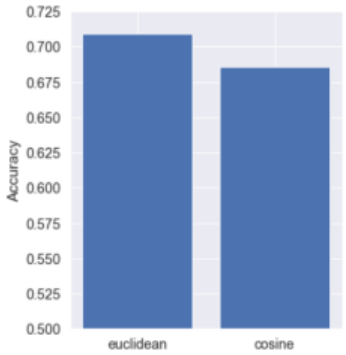

Figure 7 Training Graph with Distance Metric Parameters

Figure 7 shows a graph of the accuracy results based on the two distance metric parameters used. The graph shows that the accuracy value of training using *Euclidian Distance* is better than the accuracy value using *Cosine Distance*.

RESULTS

Data Collection Results

The result of data collection is a *tweet* that discusses Distance Learning. The tweets that were successfully retrieved were 1014 data. The results of data collection can be seen in Table 9.

Table 9 Results of Tweet Data Retrieval

Text	Created_at
I'm an introvert, I'm willing to extend distance learning and not complain	Wed Tue 01 11:51:46
Prolonged distance learning will result in learning loss in students. #ingatpesanibu #cucitangan #pakaimasker #jagajarak https://t.co/JhlxqXxwDv	Wed Tue 01 10:12:00
Conditions in rural areas are worse because distance learning does not work https://t.co/IxlbuzNLo9	Wed Tue 01 10:52:23

Results of the *Preprocessing Stage*

In the results of this *preprocessing stage, the results obtained from the* preprocessing stage that have been carried out in the previous chapter *will be discussed* . In this *preprocessing* stage, there are several processes, the processes that will be discussed include data cleaning, *case folding*, tokenization, normalization, *stopword removal*, and *stemming*. The following are the results of the process already mentioned.

Data Cleansing Results

Data cleansing is carried out on text data that has gone through the labeling stage to eliminate unnecessary components in sentiment analysis. The following is a comparison between the uncleaned data and the cleaned data can be seen in Table 10.

Table 10 Comparison of Data Before and After Data Cleansing

Before Data Cleansing	After Data Cleanup
I'm an introvert, I'm willing to extend distance learning and not complain	I'm an introvert, I'm willing to extend distance learning and not complain
Prolonged distance learning will result in learning loss in students. #ingatpesanibu #cucitangan #pakaimasker #jagajarak https://t.co/JhlxqXxwDv	Prolonged distance learning will result in learning loss in students
Conditions in rural areas are worse because distance learning does not work https://t.co/IxlbuzNLo9	Conditions in rural areas are worse because distance learning is not running

Based on the comparison in Table 10, it can be seen that the result of this data cleansing can remove *emoticons* in the first text, *hashtags* and URLs in the second text, and URLs in the third text.

Case Folding *Results*

Case folding is carried out on text data that has previously gone through a data cleaning process. The following is a comparison between data that has not gone through the *case*

folding stage and data that has gone through *the case folding* stage which can be seen in Table 11.

Table 11 Comparison of Data Before and After *Case Folding*

Before *Case Folding*	After *Case Folding*
I'm an introvert, I'm willing to extend distance learning and not complain	I'm an introvert, I'm willing to extend distance learning and not complain
Prolonged distance learning will result in learning loss in students	prolonged distance learning will result in learning loss in students
Conditions in rural areas are worse because distance learning is not running	Conditions in rural areas are worse because distance learning does not run

Based on the comparison in Table 11, it can be seen that words such as "Learning" in the second text and the word "Condition" in the third text are changed to lowercase letters in the *case folding process*.

Tokenization Results

Tokenization is carried out on text data that has previously gone through *the case folding* process. The result of this tokenization process will convert text data into chunks of words called tokens. The following is a comparison between untokenized data and tokenized data which can be seen in Table 12.

Table 12 Comparison of Data Before and After Tokenization

Before Tokenization	After Tokenization
I'm an introvert, I'm willing to extend distance learning and not complain	['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']
prolonged distance learning will result in learning loss in students	['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']
Conditions in rural areas are worse because distance learning does not run	['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']

Data Normalization Results

Data normalization is carried out on data that has previously gone through the tokenization process. The result of the normalization process will convert the text containing the abbreviated word into the original word. The following is a comparison between the unnormalized data and the normalized data which can be seen in Table 13.

Table 13 Comparison of Data Before and After Normalization

Before Normalization	After Normalization
['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']	['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']
['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']	['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']
['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']	['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']

Based on the comparison in Table 13, it can be seen that after going through the process of normalization, the third text that has an abbreviated word, namely "because", changed to the original word, namely "because".

Hasil *Stopwords Removal*

This stopwords removal is carried out on data that has gone through a normalization process. The following is a comparison between the data that has gone through *Stopwords Removal* and the data that has gone through *Stopwords Removal* can be seen in Table 14.

Table 14 Comparison of Data Before and After *Stopwords Removal*

Sebelum *Stopwords Removal*	**Sesudah *Stopwords Removal***
['I', 'introvert', 'I', 'willing', 'extend', 'learning', 'distance', 'distant', 'and', 'no', 'complain']	['introvert', 'willing', 'extending', 'learning', 'distance', 'complaining']
['learning', 'distance', 'distant', 'which', 'prolonged', 'will', 'cause', 'occurrence', 'loss', 'learning', 'on', 'student']	['learning', 'distance', 'prolonged', 'cause', 'loss', 'learning', 'student']
['condition', 'in', 'rural', 'more', 'bad', 'because', 'learning', 'distance', 'far', 'not', 'walk']	['condition', 'rural', 'poor', 'learning', 'distance', 'walking']

Based on the comparison in Table 14, it can be seen that after going through the *stopword removal process* on the first text, a number of words will be deleted such as "I", "far", "and", and "no". In the second text some of the words that are removed are "far", "will", "occurrence", and "on", while in the third text some of the words that are removed are "in", "more", "because", "far", and "not".

Voting Results

The last process in *preprocessing* is *stemming*. *Stemming* is carried out on data that has previously gone through *the stopword removal process*. The following is a comparison between data that has not been *polled* and data that has gone through *stemming* which can be seen in Table 15.

Table 15 Comparison of Data Before and After *Voting*

Before *Voting*	**After *Voting***
['introvert', 'willing', 'extending', 'learning', 'distance', 'complaining']	introverts are ready to teach distance complaining
['learning', 'distance', 'prolonged', 'cause', 'loss', 'learning', 'student']	Teaching Long Distance Due to Student Loss
['condition', 'rural', 'poor', 'learning', 'distance', 'walking']	Poor Village Conditions Teach Distance

Based on the comparison in Table 15, it can be seen that after going through the *stemming* process, some of the words that were removed from the first text were the words "willing", "extending", "learning". {There are two texts whose suffixes are "learning", "prolonged", "resulting", "loss", and "learning", while in the third text there are the words "rural", "learning" and "walking".

Word Weighting Results

The results of the word weighting from the preprocessed text sample can be seen in Table 4.8. In the results of word bottling, there are scores on several words according to the calculations that have been done in the previous chapter.

Table 16 Term *Frequency Inverse Document Frequency Results*

Term	TF-IDF		
	Text 1	**Text 2**	**Text 3**
Introvert	0,217272	0	0
Chair	0,217272	0	0
Panjang	0,187865	0,160866	0
Ajar	0,167	0,143	0,167
Distance	0,167	0,143	0,167
Complaints	0,217272	0	0
Consequences	0	0,160866	0
Lost	0	0,160866	0
Students	0	0,160866	0
Conditions	0	0	0,217272
Village	0	0	0,217272
Bad	0	0	0,217272
Roads	0	0	0,217272

Here are some words that have the highest frequency of occurrence in the word weighting process, which can be seen in Figure 4.1.

```
TF-IDF Best Scores:
dukung :  0.2981679458293734
siswa :  0.23097915815030706
langsung :  0.37794218054892276
internet :  0.2799473492102375
kuota :  0.30077561939868536
bantu :  0.26950212802495926
pandemi :  0.20035652128750095
didik :  0.21985784837416755
sektor :  0.4079009693283655
salur :  0.46751782242428186
```

Figure 8 Words with the Highest Frequency

Model Training Results

The K-Nearest Neighbor *model training* was carried out using 3 *hyperparameters*, namely k values, *weights*, and distance metrics. The results of the model training with *optimal hyperparameters* can be seen in Table 16.

Table 16 *Best Hyperparameters* in Model Training

Parameter	***Value***	**Train Accuracy (%)**
K value	22	69,67%
weights	*distance*	70,66%
Distance Metrics	*Euclidian*	70,66%

Model Evaluation

At this stage, an evaluation of the model that has been formed is carried out. The model evaluation was carried out using *the Confusion Matrix*, which is a matrix of prediction results compared to the original class of test data. The results displayed on the *Confusion Matrix* can be categorized into four types of data, including *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), and *False Negative* (FN). *True Positive* (TP) is data with a positive label that is correctly classified in the positive class. *True Negative* (TN) is data that has a negative

value and has been correctly classified in the negative class. *False Positive* (FP) is data with a positive value but is classified in a negative class. *False Negative* (FN) is data with a negative value but is classified in the positive class.

The model evaluation was carried out with 203 test data that had been labeled, consisting of 115 data labeled positive and 88 data labeled negative. The results of the *Confusion Matrix* can be seen in Table 17.

Table 17 *Matrix Confusion Table*

	Negative Sentiment Analysis Results	**Positive Sentiment Analysis Results**
Negative Original Sentiment	(*True Negative*) 66	(*False Positive*) 22
Positive Original Sentiment	(*False Negative*) 30	(*True Positive*) 85

The results of the test with *the Confusion Matrix* contained in Table 17 of the 203 test data, consisted of 151 correctly classified data and 52 incorrectly classified data. The number of data included with *True Positive* is 85 data and *True Negative* is 66 data, while the number of data included *in False Positive* is 22 data and *False Negative* is 30 data. The results of such tests can be used to calculate accuracy. Accuracy is one of the evaluations on the classification task to find out how many times data can be classified according to the class correctly. Accuracy can be calculated by comparing the number of correct classifications (*True Positive* and *True Negative*) with the overall data. The accuracy calculation can be formulated with the following equations:

$$Akurasi = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$
$$= \frac{(85 + 66)}{(85 + 30 + 22 + 66)}$$
$$= \frac{151}{203}$$
$$= 0{,}7438\ x\ 100\% = 74{,}38\%$$

According to Gorunescu (2011), the accuracy values produced in the classification process can be categorized into *ranges* where each *range* has a certain meaning. *The range* of accuracy values and their accuracy can be seen in Table 18.

Table 18 *Range* of Accuracy Values

No.	***Accuracy* Value Range**	**Arts**
1	0,90 – 1,00	*Excellent classification*
2	0,80 – 0,90	*Good Classification*
3	0,70 – 0,80	*Fair Classification*
4	0,60 – 0,70	*Poor Classification*
5	0,50 – 0,60	*Failure*

After performing the accuracy calculation, the accuracy value obtained was 0.7438 or 74.38%. The accuracy value if assessed based on Table 18 can be classified into *fair classification*.

Application Implementation

Application implementation consists of several stages, including Home page implementation, Dataset, Data Cleaning, *Case Folding*, Tokenization, Normalization, *Stopwords Removal*, *Stemming*, Test Results, and About.

1. **Home Page**

The Home page is the first page that is displayed when a user enters a *website*. This page serves as the main page of the *Distance Learning Sentiment Analysis* website.

Figure 9 Home Page View

Figure 9 shows the implementation results of the Home page. At the top of the page there is a *website* title and on the left side of the page there are 10 menus that function to move pages. This Home page contains a brief explanation of Distance Learning.

2. **Menu *Preprocessing***

This menu is a menu that has sub-sub-menus in the *preprocessing* stage of sentiment analysis. The sub-sub-menus contained in this *preprocessing* menu are as follows:

a. **Dataset Page**

The Dataset page is a page that displays datasets from Twitter social media for *training* and *testing*. On this page, there is a table that contains the *tweets* taken and the labels of each *tweet*. Figure 10 shows the implementation results of the Dataset page.

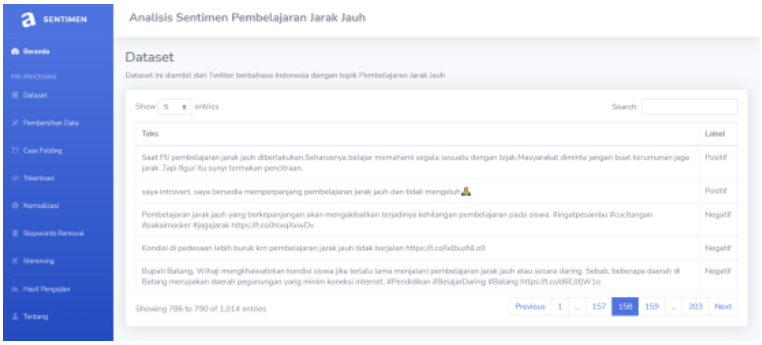

Figure 10 Dataset Page View

b. **Data Cleanup Page**

The Data Cleanup page is a page that contains *tweets* that have been cleaned to remove unnecessary components. On this page, there is a table that contains *tweet* data before

and after data cleansing. Figure 4.4 shows the implementation results of the Data Cleanup page.

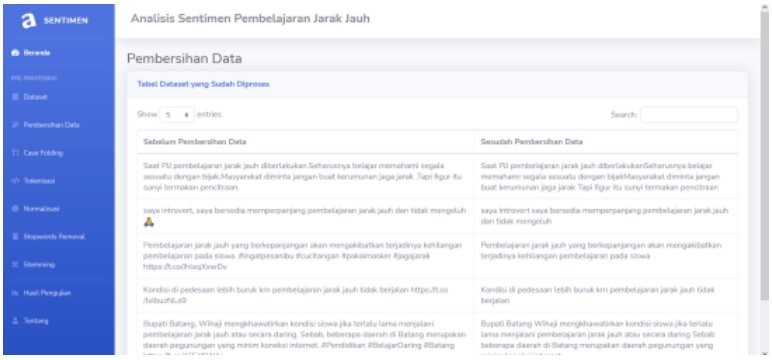

Figure 11 Data Cleanup Page View

c. **Case Folding *Page***

The Case Folding *page* is a page that contains *tweets* that have been processed to change the capital letters in sentences to lowercase letters. On this page there is a table containing tweet data before and after *Case Folding*. Figure 12 shows the implementation results of the *Case Folding page*.

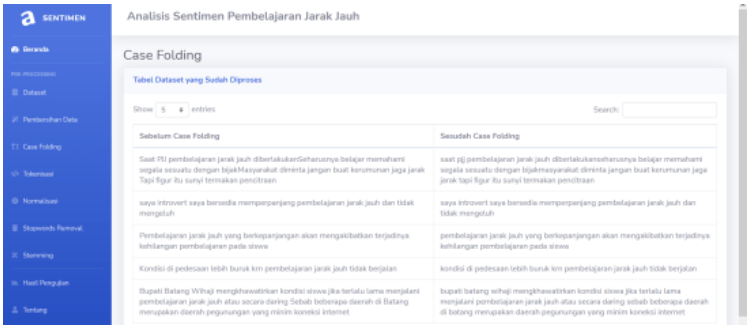

Figure 12 Case *Folding Page View*

d. **Tokenization Page**

The Tokenization page is a page that contains *tweets* that have been processed to divide sentences into token pieces. On this page there is a table containing *tweet* data before and after Tokenization. Figure 13 shows the implementation results of the Tokenization page.

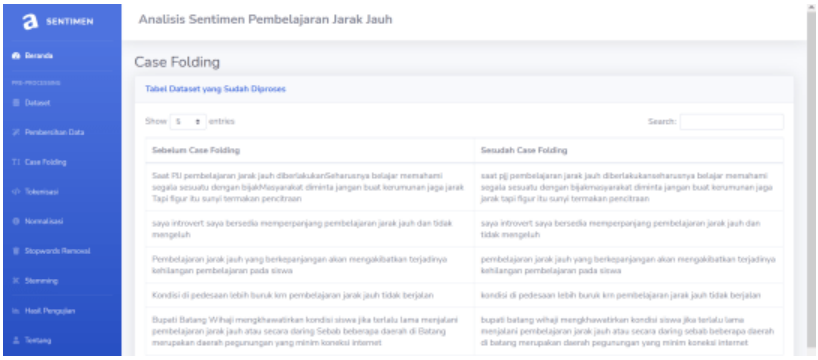

Figure 13 Tokenization Page View

e. **Normalization Page**

The Normalization page is a page that contains *tweets* that have been processed to change the abbreviated word to its original form. On this page there is a table containing *tweet* data before and after normalization. Figure 14 shows the implementation results of the Normalization page.

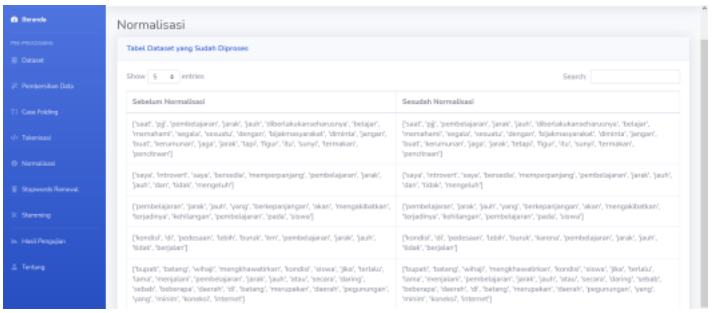

Figure 14 Normalization Page View

f. **Halaman *Stopwords Removal***

The *Stopwords Removal* page is a page that contains *tweets* that have been processed to remove words that are not needed in sentiment analysis. On this page there is a table containing *tweet* data before and after *Stopwords Removal*. Figure 4.8 shows the implementation results of the *Stopwords Removal page*.

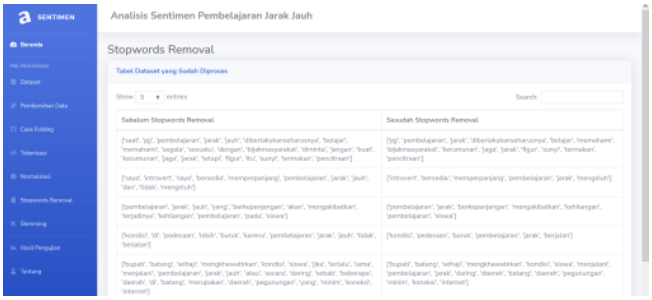

Figure 15 Stopword Removal Page View

g. **Halaman *Mood***

The *Stemming* page is a page that contains *tweets* that have been processed to change the words in the sentence to their basic form. On this page there is a table that contains *tweet* data before and after *Voting*. Figure 4.9 shows the implementation results of the *Voting page*.

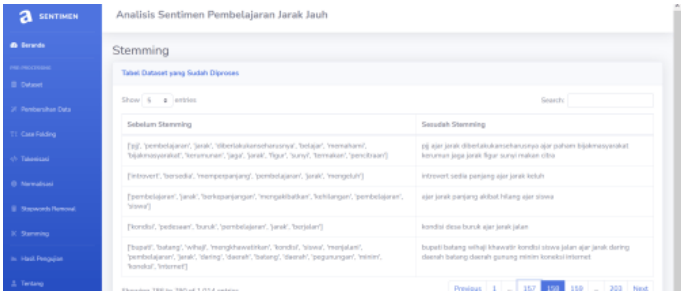

Figure 16 Stemming Page View

h. **Test Results Page**

The Test Results page is a page that displays the test results of the K-Nearest Neighbor model. This page consists of 4 tabs, namely the classification results tab, *the plot bar*, *pie chart*, and *wordcloud*. *The* tabs on this page are as follows:

1) ***Classification Results*** **Tab**

The Classification Results tab displays a table that contains tweet data, *predicted labels*, and *actual labels*. Figure 4.10 shows the implementation results of the *classification results* tab.

Figure 17 Classification *Results* Tab View

2) **Data Visualization Tab with *Plot Bar***

The second tab on the test results page will display a data visualization image in the form of *a plot bar*. Figure 18 shows the implementation results of the data visualization tab with *the Plot Bar*.

Figure 18 Data Visualization Tab View with *Plot Bar*

3) ***Data*** **Visualization Tab with *Pie Chart***

The second tab on the test results page will display a data visualization image in the form of *a plot bar*. Figure 19 shows the implementation results of the data visualization tab with *the Pie Chart*.

Figure 19 Data Visualization Tab View with *Pie Chart*

4) ***Data* Visualization Tab with *Wordcloud***

The last tab on the test results page will display a data visualization image in *wordcloud*. Data visualization in *wordcloud form* is divided into *wordcloud* for positive and negative sentiment. Figure 4.13 shows the implementation results of the data visualization tab with *Wordcloud*.

Figure 20 Data Visualization Tab View with *Wordcloud*

5) **About Page**

The About page is a page that contains information about the *website* creator. On this page there is an image that displays a photo of the *website creator* as well as the name, NPM, and class of the *website* creator. Figure 21 shows the implementation results of the Home page.

Figure 21 About Page View

CONCLUSION

Based on the results of the experiments carried out in this study, several conclusions can be drawn, including the following: The results obtained on the labeling of tweet data carried out manually on 1014 data obtained 576 positive tweet data and 438 negative tweet data. The data is then processed through the preprocessing stage, which consists of data cleaning, case folding, tokenization, data normalization, stopword removal, and stemming. The processed data can be used for sentiment analysis. Classification uses the K-Nearest Neighbor method with the most optimal hyperparameters, namely with a value of k equal to 22, weights with distance weights and distance metric with Euclidian Distance. The classification of test data yielded 107 positive data and 96 negative data. The most classification results were obtained by positive sentiment. The calculation of the accuracy value tested in this study was obtained with the Confusion Matrix. The results of data classification using the K-Nearest Neighbor method with the most optimal hyperparameters resulted in an accuracy in the test data of 74.38% which can be classified as fair classification.

REFERENCES

Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M. M., & Williams, H. E. (2007). Stemming Indonesian: A confix-stripping approach. *ACM Transactions on Asian Language Information Processing (TALIP)*, *6*(4), 1–33.

Abijono, H., Santoso, P., dan Anggreini, N. L. (2021). Algoritma Supervised Learning Dan Unsupervised Learning dalam Pengolahan Data. Jurnal Teknologi Terapan: G-Tech, 4(2), 315–318.

Berry, M. W., dan Kogan, J. (2010). Text Mining Application and Theory. United Kingdom: WILEY.

Binanto, I. (2010). Multimedia Digital-Dasar Teori dan Pengembangannya. Yogyakarta: Andi.

Chomboon, K., Chujai, P., Teerarassamee, P., Kerdprasop, K., dan Kerdprasop, N. (2015). An Empirical Study of Distance Metrics For K-Nearest Neighbor Algorithm. In Proceedings of the 3rd international conference on industrial application engineering, 280-285.

Dragut, E., Fang, F., Sistla, P., Yu, C., dan Meng, W. (2009). Stop Word and Related Problem in Web Interface Integration. California: VLDB Endowment.

Gorunescu, F. (2011). Data Mining: Concepts, models and techniques (Vol. 12). Berlin: Springer Science & Business Media.

Grover Feed, J. (2019). Perceiving Python programming paradigms. Opensource.Com. https://opensource.com/article/19/10/python-programming-paradigms

Khan, F., Kanwal, S., Alamri, S., dan Mumtaz, B. (2020). Hyper-Parameter Optimization of Classifiers, Using an Artificial Immune Network and Its Application to Software Bug Prediction. IEEE Access, 8, 20954–20964.

Liantoni, F. (2016). Klasifikasi Daun Dengan Perbaikan Fitur Citra Menggunakan Metode K-Nearest Neighbor. Jurnal ULTIMATICS, 7(2), 98–104.

Liu, B. (2010). Handbook of Natural Lenguage Processing 2nd Edition. Boca Raton: CRC Press.

Lubis, A. R., Lubis, M., dan Khowarizmi, A. (2020). Optimization of distance formula in K-Nearest Neighbor method. Bulletin of Electrical Engineering and Informatics, 9(1), 326–338.

Nurfarida, R. D., Indriati, I., dan Perdana, R. S. (2018). Klasifikasi Kemacetan Lalu Lintas di Kota Malang Pada Sosial Media Twitter Menggunakan Metode Improved K-Nearest Neighbor. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer, 3(2), 1235–1242.

Prasetyo, E. (2012). Data mining konsep dan aplikasi menggunakan matlab. Yogyakarta: Andi.

Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

Ranatarisza, M. M., dan Noor, M. A. (2013). Sistem Informasi Akuntansi pada Aplikasi Administrasi Bisnis. Malang: Universitas Brawijaya Press.

Rezwanul, M., Ali, A., dan Rahman, A. (2017). Sentiment Analysis on Twitter Data using KNN and SVM. International Journal of Advanced Computer Science and Applications, 8(6), 19–25.

Septian, J. A., Fachrudin, T. M., dan Nugroho, A. (2019). Analisis Sentimen Pengguna Twitter Terhadap Polemik Persepakbolaan Indonesia Menggunakan Pembobotan TF-IDF dan K-Nearest Neighbor. Journal of Intelligent System and Computation, 1(1), 43–49.

Sudira, H., Diar, A. L., dan Ruldeviyani, Y. (2019). Instagram Sentiment Analysis with Naive Bayes and KNN: Exploring Customer Satisfaction of Digital Payment Services in Indonesia. 2019 International Workshop on Big Data and Information Security, IWBIS 2019, 21–26.

Weiss, S. M., Indurkhya, N., Zhang, T., dan Damerau, F. (2010). Text mining: predictive methods for analyzing unstructured information. Berlin: Springer Science & Business Media