CHAPTER 4

INITIAL RESULTS

4.1 Introduction

This chapter discusses the results and sentiment analysis of the free meal program. This chapter begins with the identification of the dataset, and continues with the results of calculating the proportion of data, creating models and implementing models using machine learning techniques. The machine learning techniques used are K-nearest neighbors (KNN), Naive Bayes and Support Vector Machine (SVM). Based on the results of the implementation of these machine learning techniques, it was found that the KKN and Naive Bayes techniques had a higher percentage of accuracy and classification results compared to SVM. Details of the results and analysis are presented in the following subsections.

4.2 Exploratory Data Analysis (EDA)

Exploratory Data Analysis is very important to do before the modeling stage. Exploratory Data Analysis (EDA) can be briefly interpreted as a process of understanding data to obtain as much information as possible. In addition, EDA can also be done to understand data patterns. The `full_text` column describes the public's reaction on social media X to the free meal program. Then the reaction will be analyzed to obtain the results of sentiment analysis of the program whether it is positive, negative or neutral.

4.2.1 Data Collection

The data collection process was carried out using the crawling method using the Python programming language to retrieve data from the social media application X (formerly known as Twitter). The data that was successfully obtained included various tweets related to the free lunch program which was the focus of the study. After the data was collected, the data was saved in CSV file format to facilitate further analysis. The data that had been stored then went through the pre-processing and data cleaning stages, where steps such as removing URLs, user tags (@username), hashtags, numbers, punctuation, and extra spaces were carried out. In addition, stemming was also carried out to return words to their basic form and translation of English words into Indonesian to ensure the data was clean, consistent, and ready for further sentiment analysis. Figure 3.1 illustrates the flowchart process for crawling data.

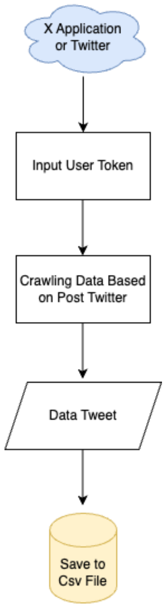
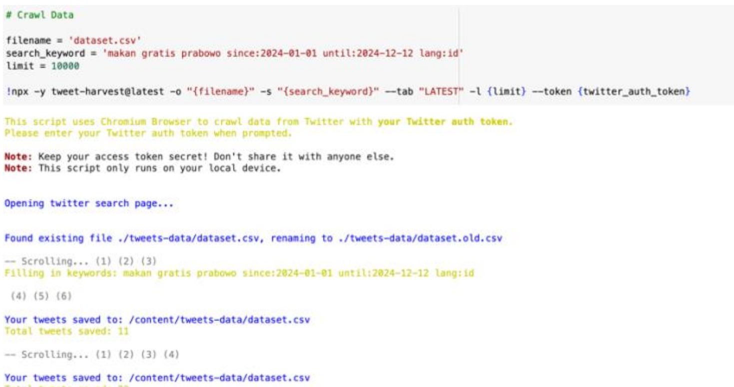


Figure 4.1 Flowchart of data crawling process from Social Media Application X (Twitter)

In Figure 4.2 is the data crawling process using Python syntax and utilizing user tweet posts.



```
# Crawl Data

filename = 'dataset.csv'
search_keyword = 'makan gratis prabowo since:2024-01-01 until:2024-12-12 lang:id'
limit = 10000

!npx -y tweet-harvest@latest -o "{filename}" -s "{search_keyword}" --tab "LATEST" -l {limit} --token {twitter_auth_token}

This script uses Chromium Browser to crawl data from Twitter with your Twitter auth token.
Please enter your Twitter auth token when prompted.

Note: Keep your access token secret! Don't share it with anyone else.
Note: This script only runs on your local device.

Opening twitter search page...

Found existing file ./tweets-data/dataset.csv, renaming to ./tweets-data/dataset.old.csv

-- Scrolling... (1) (2) (3)
Filling in keywords: makan gratis prabowo since:2024-01-01 until:2024-12-12 lang:id

 (4) (5) (6)

Your tweets saved to: /content/tweets-data/dataset.csv
Total tweets saved: 11

-- Scrolling... (1) (2) (3) (4)

Your tweets saved to: /content/tweets-data/dataset.csv
```

Figure 4.2 Process Crawling for Collection Data

In this figure 4.3 is after the data crawling process is successful, then the data will be saved in a file with csv format. The library used for this process is the Pandas library in python. After that the data that has been saved will be displayed in the form of a data frame.

```python
import pandas as pd

# Specify the path to your CSV file
file_path = f"tweets-data/{filename}"

# Read the CSV file into a pandas DataFrame
df = pd.read_csv(file_path, delimiter=",")

# Display the DataFrame
display(df)
```

	conversation_id_str	created_at	favorite_count	full_text	id_str	image_url	in_reply_to_screen_name	lang	location	quote_count	re
0	1866872716541972550	Wed Dec 11 23:25:45 +0000 2024	4	@pramudyawdynto Target makan gratis e prabowo	1866987767919677789	NaN	pramudyawdynto	in	Vladivostok, Russia	0	
1	1866464297657975123	Wed Dec 11 21:52:21 +0000 2024	1	@Yudhi2024 @prabowo masih enak dan bergizi mak...	1866964263254032437	NaN	Yudhi2024	in	NaN	0	
2	1866464297657975123	Wed Dec 11 21:21:34 +0000 2024	0	@StevanFirman15 @prabowo Yakin pak makan grati...	1866956515414094156	NaN	StevanFirman15	in	Bekasi Barat, Indonesia	0	
3	1866754957552259364	Wed Dec 11 18:42:05 +0000 2024	0	@03__nakula Makanya program Prabowo makan sian...	1866916381872230896	NaN	03__nakula	in	Bandung, Jawa Barat	0	
4	1866877557489926164	Wed Dec 11 16:07:49 +0000 2024	0	Prancis Dukung Program Makan Bergizi Gratis Pr...	1866877557489926164	NaN	NaN	in	Jakarta, Indonesia	0	

Figure 4.3 Saving and Displaying Data After Data Collection Process

The total data collected is 2916 and 15 columns. The data will be subjected to data pre-processing and data cleaning, so that it will get better and more accurate data.

4.2.2 Data Preparation and Cleaning

Data cleaning is an important process in sentiment analysis, especially to ensure that the data used is clean, relevant, and can be processed well by the model. Here are the data cleaning steps carried out on the Twitter tweet dataset about the Prabowo-Gibran free meal program : Initialize Sastrawi Stemmer, Convert text to Lowercase, RemoveURLs, Remove @username, Remove #Hastags, remove numbers, remove punctuation, Extra whitespace, Apply Stemming and Translate data to minimize English word. In figure 4.4 is the python syntax used for data pre-processing.

```python
# Preprocessing function for tweets
def preprocess_text(text):
    text = text.lower()  # Convert text to lowercase
    text = re.sub(r"http\S+|www\S+|https\S+", '', text, flags=re.MULTILINE)  # Remove URLs
    text = re.sub(r'@\w+', '', text)  # Remove @username
    text = re.sub(r'#\w+', '', text)  # Remove hashtags
    text = re.sub(r'\d+', '', text)  # Remove numbers
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra whitespace
    text = stemmer.stem(text)  # Apply stemming
    return text

# Apply preprocessing
data['full_text'] = data['full_text'].apply(preprocess_text)
```

Figure 4.4 Syntax for pre-processing data

4.2.3 Demographic and Distribution Data

Demography is the study of human populations and changes in their quantity as they relate to migration, fertility, and mortality. The term demography is derived from the Greek word and means "describing people." Thus, this discipline deals with the characteristics of populations by considering features such as sex ratio, age structure, composition, spatial distribution, and population density. (Klimczuk, 2021)

In this project, demographic data consists of user data and also user tweet posts related to the free meal program from President Prabowo - Gibran. The following is an explanation of the data distribution in the free food program dataset as shown in Figure 4.5.

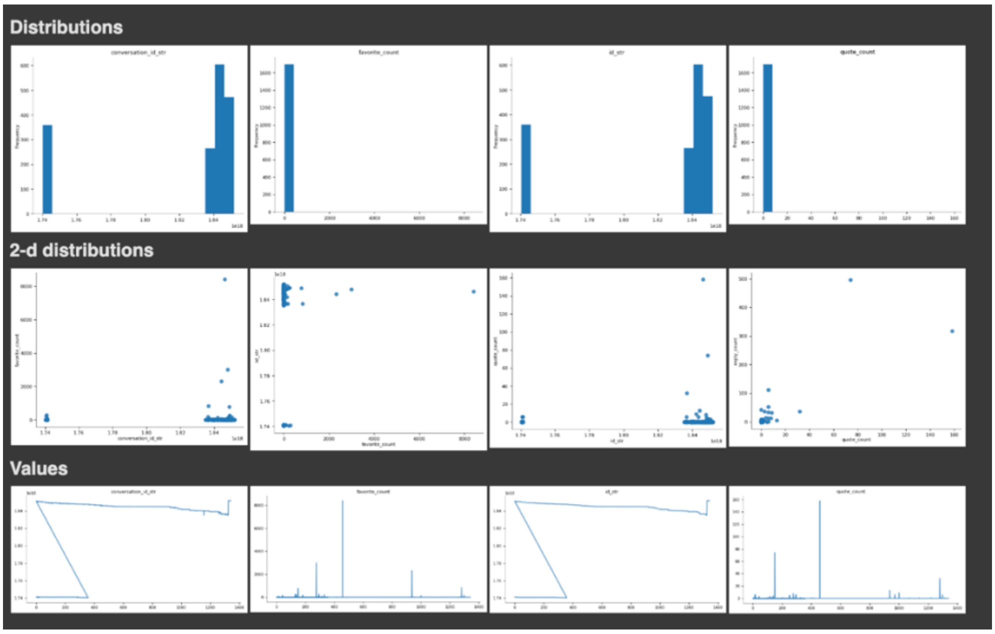


Figure 4.5 Distribution of data in each column in the dataset

Table 4.1 explains the distribution of data related to each item value in the dataset.

Table 4.1 Analysis of each column in the dataset

Distributions This section shows the distribution of values for each column of the dataset.	
`conversation_id_str`	This distribution shows unique conversation IDs that are mostly distributed in a certain range. Large ID values indicate that this is data taken from Twitter, as IDs are usually long numbers.
`favorite_count`	Distribution of the number of "likes" or "favorites" on tweets. Most tweets have a low "like" value (close to zero), indicating that many tweets receive little attention or interaction.
`id_str`	Like `conversation_id_str`, this is a unique ID for a tweet. Its distribution follows a similar long ID pattern.
`quote_count`	Distribution of the number of "quote retweets". Most of the data has a value of zero, indicating that most tweets are not quoted by

	other users. However, there are some extreme values with higher "quote" numbers.
2-d Distributions This section shows the relationship between variables with a 2-dimensional distribution.	
`favorite_count vs conversation_id_str`	This graph shows that the number of "likes" is sporadically distributed across the conversation IDs. Most of the "like" values are low, with a few outliers having high "like" counts.
`favorite_count vs id_str`	Similar to the previous relationship, but focused on the unique ID of each tweet. The pattern is similar, with a few dots indicating popular tweets.
`quote_count vs id_str`	Most tweets have a low quote value, but there are a few outliers where tweets have a significant number of quotes. This suggests that only a small number of tweets attract the attention of other users to re-comment.
Values This section visualizes the distribution of values in the form of a line:	
`conversation_id_str`	The lines indicate sequential IDs. This confirms that the data may have been collected chronologically.
`favorite_count`	The distribution pattern shows that most values are close to zero with a few peaks (outliers).
`quote_count`	Most of the values are close to zero, indicating tweets that are rarely requoted, but there are a few peaks with higher values.

In Figure 4.6 is each column in the dataset and also the data type used. It can be seen that all columns are non-null, consisting of 8 objects and 7 int64.

```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2916 entries, 0 to 2915
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   conversation_id_str     2916 non-null   int64 
 1   created_at              2916 non-null   object
 2   favorite_count          2916 non-null   int64 
 3   full_text               2916 non-null   object
 4   id_str                  2916 non-null   int64 
 5   image_url               692 non-null    object
 6   in_reply_to_screen_name 1002 non-null   object
 7   lang                    2916 non-null   object
 8   location                1352 non-null   object
 9   quote_count             2916 non-null   int64 
 10  reply_count             2916 non-null   int64 
 11  retweet_count           2916 non-null   int64 
 12  tweet_url               2916 non-null   object
 13  user_id_str             2916 non-null   int64 
 14  username                2916 non-null   object
dtypes: int64(7), object(8)
memory usage: 341.8+ KB

data.columns

Index(['conversation_id_str', 'created_at', 'favorite_count', 'full_text',
       'id_str', 'image_url', 'in_reply_to_screen_name', 'lang', 'location',
       'quote_count', 'reply_count', 'retweet_count', 'tweet_url',
       'user_id_str', 'username'],
      dtype='object')
```

Figure 4.6 Dataset Information

In Figure 4.7 Dataset Description, there are extreme values (outliers) in `favorite_count`, `quote_count`, `reply_count`, and `retweet_count` indicating that some tweets are very viral, which may be caused by content factors or accounts with many followers.

```
data.describe()
```

	conversation_id_str	favorite_count	id_str	quote_count	reply_count	retweet_count	user_id_str
count	2.916000e+03	2916.000000	2.916000e+03	2916.000000	2916.000000	2916.000000	2.916000e+03
mean	1.839460e+18	13.176269	1.839570e+18	1.172154	2.895405	3.250000	1.219714e+18
std	3.919510e+16	210.665196	3.919693e+16	30.272856	43.586168	61.184726	6.873334e+17
min	1.740213e+18	0.000000	1.740399e+18	0.000000	0.000000	0.000000	1.538445e+07
25%	1.843504e+18	0.000000	1.843553e+18	0.000000	0.000000	0.000000	8.646796e+17
50%	1.848229e+18	0.000000	1.848231e+18	0.000000	0.000000	0.000000	1.598136e+18
75%	1.862699e+18	0.000000	1.862782e+18	0.000000	0.000000	0.000000	1.696032e+18
max	1.877502e+18	8417.000000	1.877504e+18	1353.000000	1708.000000	2966.000000	1.865041e+18

Figure 4.7 Dataset Description

4.2.4 Data Proportion

Proportion data is used to help understand the balance of data between relevant categories (Normal) and categories that can be considered noise (Bot). Duplicated tweets (Bot) are often less relevant for sentiment analysis because they tend not to represent the user's original opinion.

Some of the reasons why proportion data is needed include for the process of Data Quality Identification, Data Cleaning, Sentiment Model Evaluation and Better Decision Making. In the Figure 4.8 it can be seen that the data categories are divided into two, namely Bot and Normal. The distribution of the proportion of each of these data is Bot, there are 14.09% of the total dataset or around 411 data, while the Normal category is the majority of the data, which is around 85.91% or 2505 data. In this case, it can be seen that if the proportion of Bot is too large, the sentiment model can be biased because duplicate tweets have a repetitive pattern and do not reflect real opinions. In this case, the proportion of Bot is 14.09%, which is still considered reasonable to overcome.

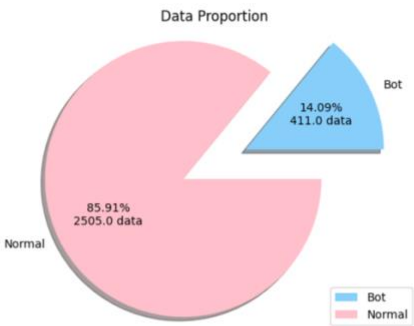


Figure 4.8 Data Proportion

From the proportion data results, it was found that:

a. Balanced Distribution

A much larger proportion of the Normal category (82.66%) indicates that the majority of data comes from unique sources, so this dataset is valid enough to be used in sentiment analysis.

b. Bot Influence

Although the number of Bots is small, this data still needs to be considered because it can affect the final results of the analysis if not filtered properly.

c. Initial Conclusion

This graph shows that the dataset is of sufficient quality because most of the data are unique tweets (Normal), but preprocessing steps to eliminate duplication (Bot) are still needed before further analysis.

4.3 Sentiment Analysis

In this sentiment analysis section, we will identify each word from social media tweets and categorize whether the word is positive, negative or neutral. Here are some examples of positive, negative and neutral sentences.

Table 4.2 Some Examples of Sentiment Analysis sentences

Full_text	**Sentiment**
sindir keras mahfud md soal program makan sian..	Positif
saltingan banget dia kenapa sihhh dari dulu ga..	Neutral
banyak yang belum sadar bahwa presiden saat in...	Negative

The Figure 4.9 shows a word cloud for positive sentiment reviews. The word cloud analysis illustrates that “healthy”, “economic movement”, “prosperous”, “thank you”, “future”, “lunch”, “free lunch”, “industrial sector” and “great potential” are the most frequently used words in the reviews. The word “Lunch” is the most frequently used word, indicating that this program often receives positive reviews from public reactions on social media. While the words “healthy” and “prosperous” are the words that are the hopes of this free meal program when it is run by the government under the leadership of President and Vice President Prabowo Subianto and Gibran.

Figure 4.9 World Cloud of Positive Sentiment

The Figure 4.10 shows a word cloud for negative sentiment reviews. The word cloud analysis illustrates that "no", "prabowo", "problem", "fail", "stupid", "sarcasm", "poor", "corruptor" and "politics" are the most frequently used words in the reviews. The words "fail", "problem" and "poor" illustrate that many people will be pessimistic about this free meal program. They assume that this program will fail and not continue and will not be on target. So it will add new problems for the country, while currently those who need to be helped are people with poor economies to be more prosperous.

Figure 4.10 World Cloud of Negative Sentiment

The Figure 4.11 shows a word cloud for neutral sentiment reviews. The word cloud analysis illustrates that “nutrition”, “confused”, “program”, “campaign promise”, “help”, “realization”, “children”, “economy” and “need” are the most frequently used words in the reviews. These words may indicate that the lunch program provides a little hope for children to get better nutrition. In addition, the community also hopes that the government can realize the program, not just make promises during the campaign.

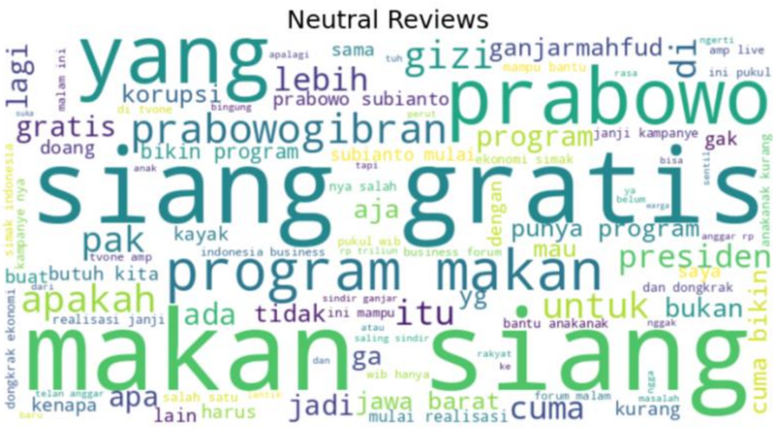


Figure 4.11 World Cloud of Neutral Sentiment

From the figure 4.12 and 4.13 the results of the sentiment analysis above, it can be seen that the sentiment results are positive. With a total of 95.57% data or 1624 data. Then the neutral data is 2.82% and the least is data with negative sentiment which is only 1.7%. Thus, we can conclude that the results of the sentiment analysis related to the free meal program are positive and can be interpreted as meaning that the public agrees to the free meal program.

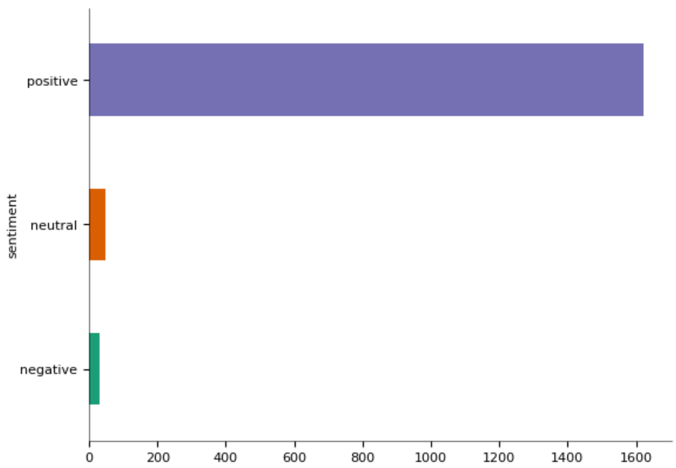


Figure 4.12 Distribution of sentiment analysis result categories

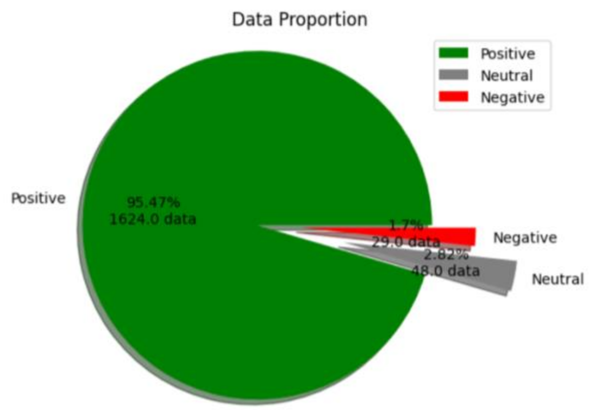


Figure 4.13 Graph of Proportion of data from sentiment analysis results

4.4 Feature Extraction

The feature extraction process is carried out to convert raw text data into numeric representations that can be processed by the machine learning model. The first step is to perform class balancing by ensuring that the amount of data for each sentiment class (positive, neutral, and negative) has the same proportion. This aims to avoid model bias towards classes with dominant data amounts. In this implementation, the under sampling method is applied, namely selecting the same number of samples based on the smallest number of classes, resulting in a balanced distribution with each class having 91 samples.

The next step, the text data is converted into a vector using the TF-IDF (Term Frequency-Inverse Document Frequency) Vectorizer, with a maximum feature limit used of 5,000 features. TF-IDF calculates the weight value of each word based on its frequency of occurrence in a particular document (term frequency) and how unique the word is compared to other documents (inverse document frequency). This process produces a matrix with a size of (273, 1736), where 273 is the number of documents (data) after balancing, and 1736 is the number of unique features generated.

The last step is to encode the sentiment label using `LabelEncoder`. The sentiment labels “positive,” “neutral,” and “negative” are transformed into their respective numeric values (0 for negative, 1 for neutral, and 2 for positive). This process ensures that the target data conforms to a format acceptable to the machine learning algorithm. This transformation produces data ready for the model training process, with numeric representations of the text and encoded sentiment labels.

4.5 Model Development

At this stage, we will help the model become Model X and Y. And then use the TF-IDF Vectorizer (Term Frequency-Inverse Document Frequency Vectorizer) technique to convert text data into numeric representations (vectors) before the data is processed using machine learning algorithms. Its main function is to give weight to each word in the document, so that relevant words are more prominent and common words are less influential. TF-IDF Vectorizer is an important step in the text-based machine learning pipeline. It helps capture important information from text data and ignores irrelevant information, thereby improving the performance and accuracy of the prediction model.

The Importance of TF-IDF in Machine Learning :

a) Reducing Data Dimensionality: TF-IDF allows the use of only relevant words (for example, with `max_features=5000`) without having to process all the words in the dataset, making the model more efficient.

b) Reducing Overfitting: Very common or irrelevant words (stopwords) are given low weight or ignored, which helps the model not to be affected by noise.

c) Highlight Relevant Words: Words that are specific and relevant to a particular class will get higher weights, increasing the model’s accuracy in understanding the relationship between words and target labels.

d) Compatible with Machine Learning: Machine learning models like SVM, Naive Bayes, or KNN can only process numeric data. TF-IDF converts text data into a numeric format that is acceptable to the model.

In Figure 4.14 is the code syntax for the model creation process and also the implementation of the TF-IDF Vectorizer technique for each model.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# Divide the data into features (X) and targets (y)
X = data['full_text']
y = data['sentiment']
```

```python
# TF-IDF Vectorizer to convert text to vectors
vectorizer = TfidfVectorizer(max_features=5000)  # Adjust max_features as needed
X_vectorized = vectorizer.fit_transform(X)
```

```python
# Splitting data into training and test data
X_train, X_test, y_train, y_test = train_test_split(X_vectorized, y, test_size=0.2, random_state=42)
```

Figure 4.14 Creating Data Model and Implement TF-IDF

4.6 Model Evaluation and Implement Machine Learning Technique

The model that has been successfully created previously will then be processed for implementation in machine learning techniques. To get better accuracy, in this project researchers will use several machine learning techniques such as KNN (K-Nearest Neighbors), Naive Bayes, and SVM (Support Vector Machine). In this process, Hyperparameter Tuning will be used for each machine learning technique. Hyperparameter tuning is done to find the best parameters of the three machine learning models. Here are the details:

1) KNN (K-Nearest Neighbors)

   Tested parameters:

   a) `n_neighbors`: Number of neighbors considered (3, 5, 7, 9).

   b) weights: How to give weight to neighbors (uniform for all neighbors have the same weight, and distance for weights based on distance).

   The best model is stored in the variable `knn_best_model`.

2) Naive Bayes

   Tested parameters:

   alpha: Smoothing parameter (tested values: 0.1, 0.5, 1.0, 1.5, 2.0).

   The best model is stored in the variable `nb_best_model`.

3) SVM (Support Vector Machine)

   Tested parameters:

   C: Regularization parameter (tested values: 0.1, 1, 10, 100).

kernel: Kernel function (linear for linear kernel, and rbf for radial basis function kernel). The best model is stored in the variable `svm_best_model`.

After finding the best model for each algorithm, an evaluation is carried out using test data (`X_test` and `y_test`) with the following steps:

a) Test Data Prediction

KNN: Using `knn_best_model` to predict.

Naive Bayes: Using `nb_best_model` to predict.

SVM: Using `svm_best_model` to predict.

b) Calculating Accuracy

Accuracy is calculated with the accuracy score function, which compares the predictions to the original labels in the test data.

After conducting model evaluation and data test, the results were obtained as:

KNN : 94.34%

Naive Bayes : 94.00%

SVM : 94.34%

```
# Predicting test data and calculating accuracy
knn_pred = knn_best_model.predict(X_test)
nb_pred = nb_best_model.predict(X_test)
svm_pred = svm_best_model.predict(X_test)

# Displays accuracy results and classification reports
print("KNN Accuracy:", accuracy_score(y_test, knn_pred))

print("\nNaive Bayes Accuracy:", accuracy_score(y_test, nb_pred))

print("\nSVM Accuracy:", accuracy_score(y_test, svm_pred))
```

```
KNN Accuracy: 0.9434931506849316

Naive Bayes Accuracy: 0.940068493150685

SVM Accuracy: 0.9434931506849316
```

Figure 4.15 Accuracy Results and Classification Reports

4.7 Summary

After conducting several processes for sentiment analysis, model evaluation, and data testing with machine learning data techniques, it was found that the sentiment obtained was positive with a total of 95.47% or 1624 data. And for the accuracy of the machine learning technique, KNN and SVM have the same high accuracy (94.35%) and are better than Naïve Bayes (94.00%) in predicting sentiment. Hyperparameter tuning successfully improves model performance by selecting the best combination of parameters for each algorithm. All models show very good performance with accuracy above 94%, which means that the data has been processed well (for example: through TF-IDF Vectorizer and balanced training-test data division). Although Naïve bayes is slightly lower in accuracy, this algorithm is usually more stable for data with high dimensions and complex distributions, so it can be considered for larger or more varied datasets. With these results, KNN or SVM can be selected as the best model for sentiment analysis cases based on performance on test data.

REFERENCES

Klimczuk, A. (2021). Introductory Chapter: Demographic Analysis. In *Demographic Analysis - Selected Concepts, Tools, and Applications*. IntechOpen. https://doi.org/10.5772/intechopen.100503