CHAPTER 4

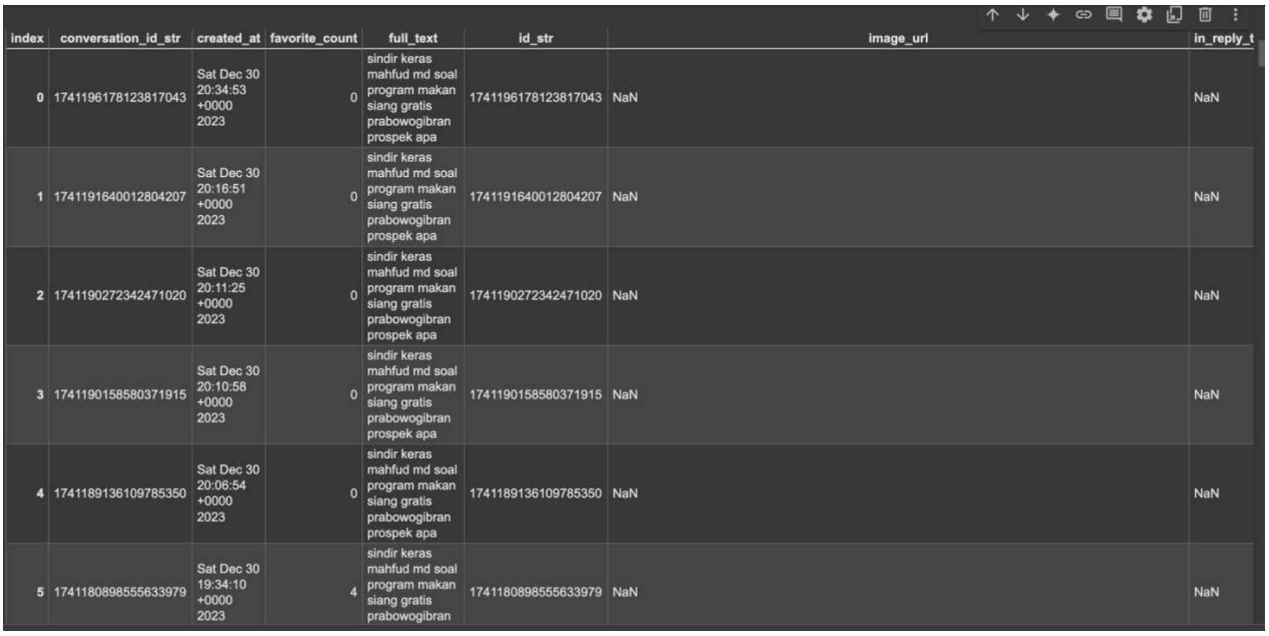
INITIAL RESULTS

4.1 Overview

This chapter discusses the results and sentiment analysis of the free meal program. This chapter begins with the identification of the data set, 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.

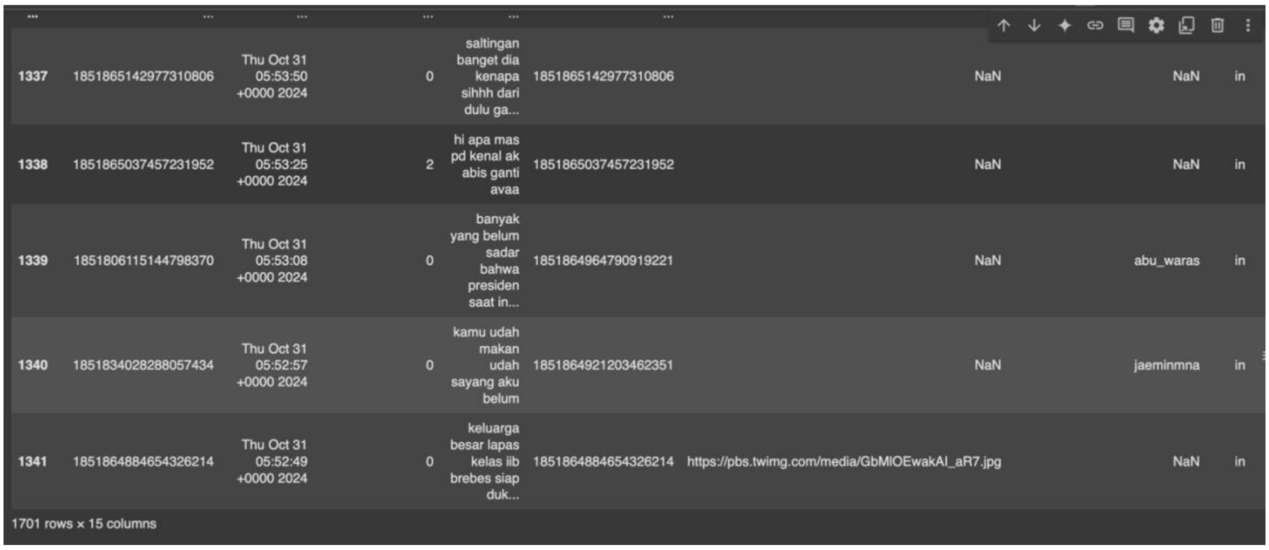
4.2 Exploratory Data Analysis (EDA)

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.



index	conversation_id_str	created_at	favorite_count	full_text	id_str	image_url	in_reply_t
0	1741196178123817043	Sat Dec 30 20:34:53 +0000 2023	0	sindir keras mahfud md soal program makan siang gratis prabowogibran prospek apa	1741196178123817043	NaN	NaN
1	1741191640012804207	Sat Dec 30 20:16:51 +0000 2023	0	sindir keras mahfud md soal program makan siang gratis prabowogibran prospek apa	1741191640012804207	NaN	NaN
2	1741190272342471020	Sat Dec 30 20:11:25 +0000 2023	0	sindir keras mahfud md soal program makan siang gratis prabowogibran prospek apa	1741190272342471020	NaN	NaN
3	1741190158580371915	Sat Dec 30 20:10:58 +0000 2023	0	sindir keras mahfud md soal program makan siang gratis prabowogibran prospek apa	1741190158580371915	NaN	NaN
4	1741189136109785350	Sat Dec 30 20:06:54 +0000 2023	0	sindir keras mahfud md soal program makan siang gratis prabowogibran prospek apa	1741189136109785350	NaN	NaN
5	1741180898555633979	Sat Dec 30 19:34:10 +0000 2023	4	sindir keras mahfud md soal program makan siang gratis prabowogibran	1741180898555633979	NaN	NaN

Figure 4.1 Dataset



...	...	...	...	...	...			
1337	1851865142977310806	Thu Oct 31 05:53:50 +0000 2024	0	saltingan banget dia kenapa sihhh dari dulu ga...	1851865142977310806	NaN	NaN	in
1338	1851865037457231952	Thu Oct 31 05:53:25 +0000 2024	2	hi apa mas pd kenal ak abis ganti avaa	1851865037457231952	NaN	NaN	in
1339	1851806115144798370	Thu Oct 31 05:53:08 +0000 2024	0	banyak yang belum sadar bahwa presiden saat in...	1851864964790919221	NaN	abu_waras	in
1340	1851834028288057434	Thu Oct 31 05:52:57 +0000 2024	0	kamu udah makan udah sayang aku belum	1851864921203462351	NaN	jaeminmna	in
1341	1851864884654326214	Thu Oct 31 05:52:49 +0000 2024	0	keluarga besar lapas kelas iib brebes siap duk...	1851864884654326214	https://pbs.twimg.com/media/GbMIOEwakAI_aR7.jpg	NaN	in

1701 rows × 15 columns

Figure 4.2 Dataset

In Figures 4.1 and 4.2 above, it is a picture of the dataset owned after the data merging process from 2023 and 2024. The total data rows are 1701 and the columns are 15.

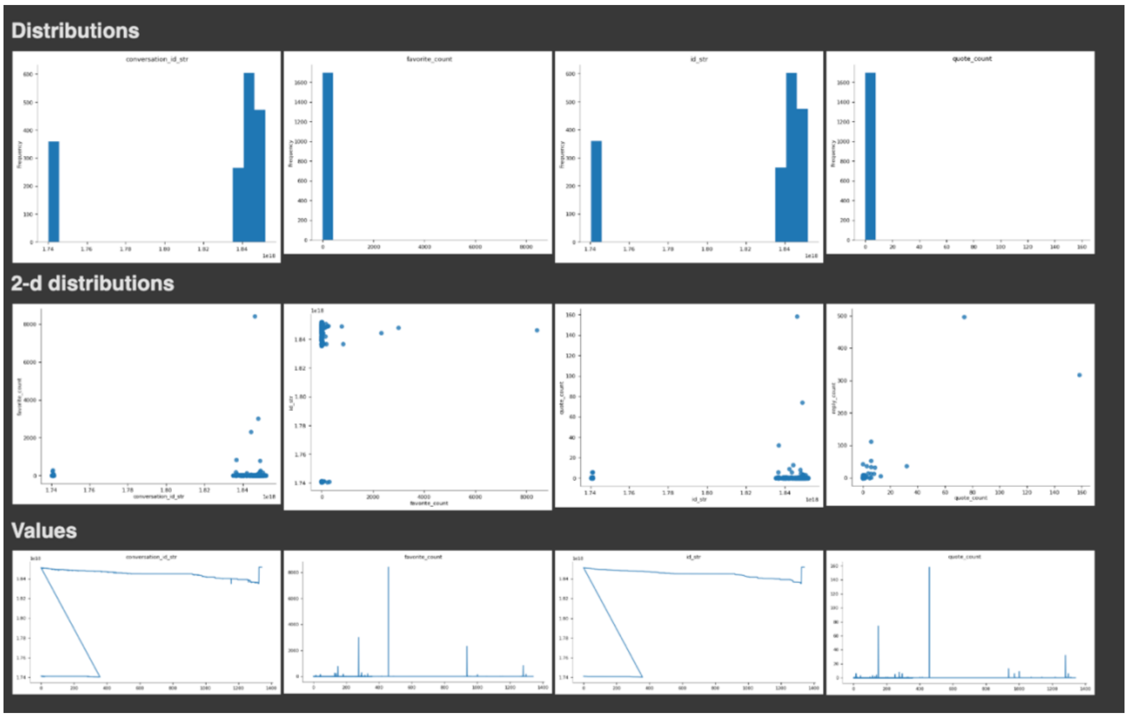


Figure 4.3 Distribution of data in each column in the dataset

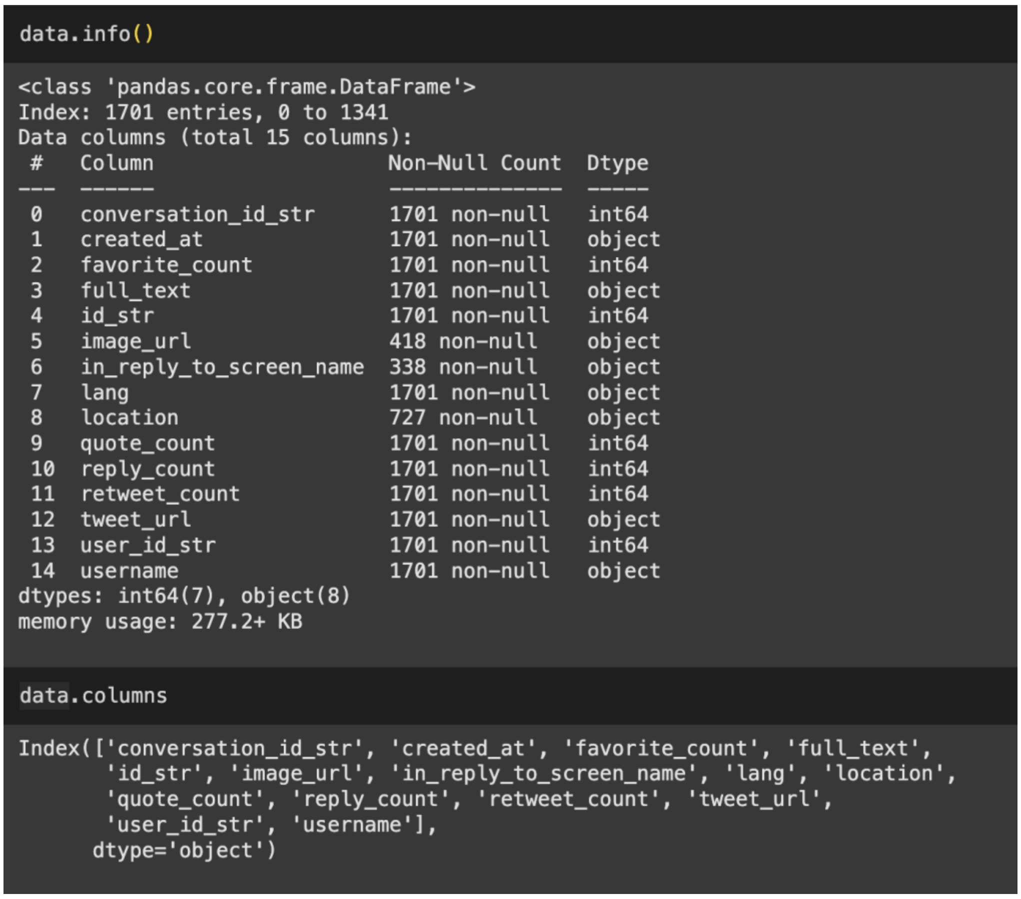
The following is an explanation of the data distribution in the free meal program dataset as shown in Figure 4.3.

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.

Table 4.1 Analysis of each column in the dataset

In Figure 4.6 below are 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.



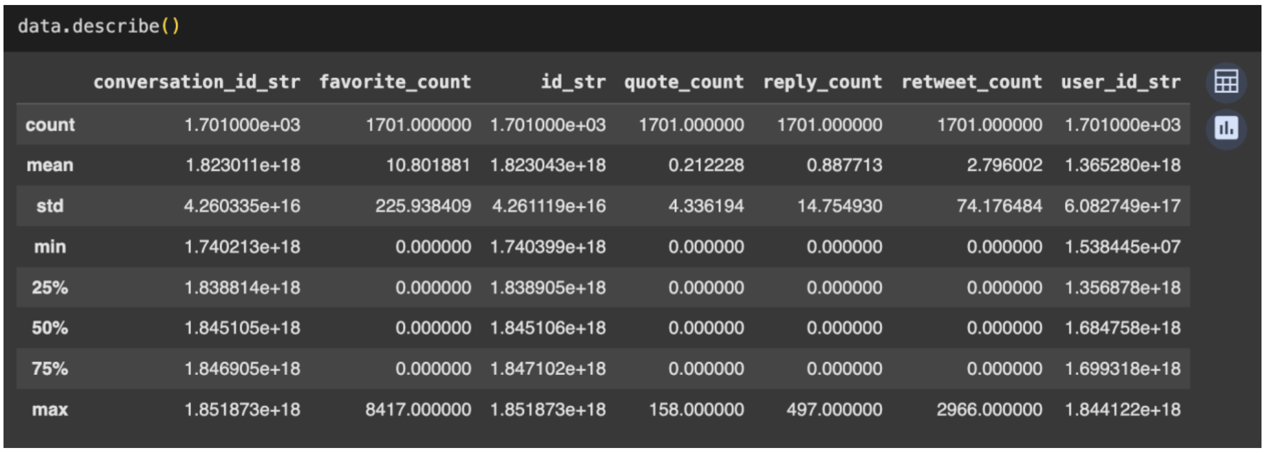
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1701 entries, 0 to 1341
Data columns (total 15 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   conversation_id_str      1701 non-null   int64 
 1   created_at               1701 non-null   object
 2   favorite_count           1701 non-null   int64 
 3   full_text                1701 non-null   object
 4   id_str                   1701 non-null   int64 
 5   image_url                418 non-null    object
 6   in_reply_to_screen_name  338 non-null    object
 7   lang                     1701 non-null   object
 8   location                 727 non-null    object
 9   quote_count              1701 non-null   int64 
 10  reply_count              1701 non-null   int64 
 11  retweet_count            1701 non-null   int64 
 12  tweet_url                1701 non-null   object
 13  user_id_str              1701 non-null   int64 
 14  username                 1701 non-null   object
dtypes: int64(7), object(8)
memory usage: 277.2+ 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 4.5 Dataset Information



```
data.describe()
```

	conversation_id_str	favorite_count	id_str	quote_count	reply_count	retweet_count	user_id_str
count	1.701000e+03	1701.000000	1.701000e+03	1701.000000	1701.000000	1701.000000	1.701000e+03
mean	1.823011e+18	10.801881	1.823043e+18	0.212228	0.887713	2.796002	1.365280e+18
std	4.260335e+16	225.938409	4.261119e+16	4.336194	14.754930	74.176484	6.082749e+17
min	1.740213e+18	0.000000	1.740399e+18	0.000000	0.000000	0.000000	1.538445e+07
25%	1.838814e+18	0.000000	1.838905e+18	0.000000	0.000000	0.000000	1.356878e+18
50%	1.845105e+18	0.000000	1.845106e+18	0.000000	0.000000	0.000000	1.684758e+18
75%	1.846905e+18	0.000000	1.847102e+18	0.000000	0.000000	0.000000	1.699318e+18
max	1.851873e+18	8417.000000	1.851873e+18	158.000000	497.000000	2966.000000	1.844122e+18

Figure 4.6 Dataset Description

In Figure 4.6 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.

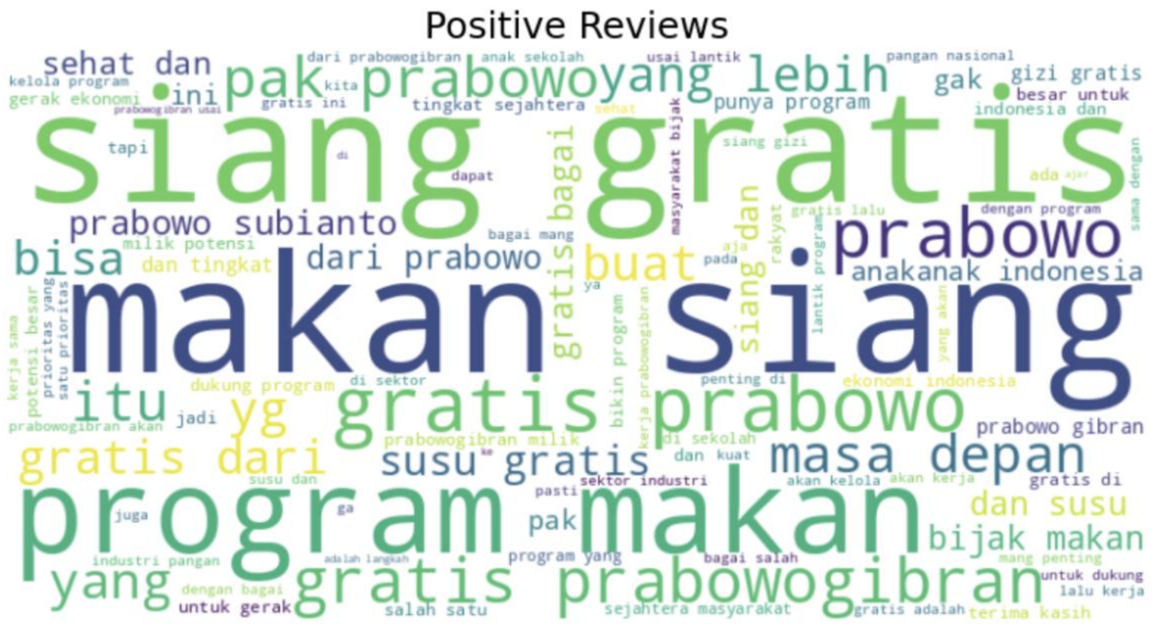


Figure 4.7 World Cloud of Positive Sentiment

The Figure 4.7 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.

The image below 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.

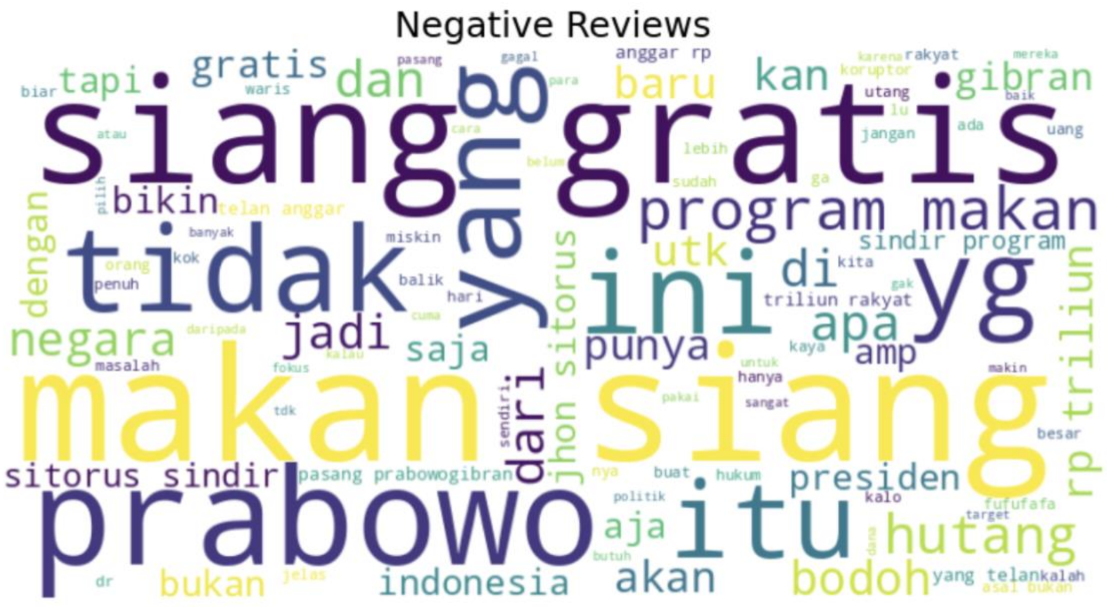


Figure 4.8 World Cloud of Negative Sentiment

The image below 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.

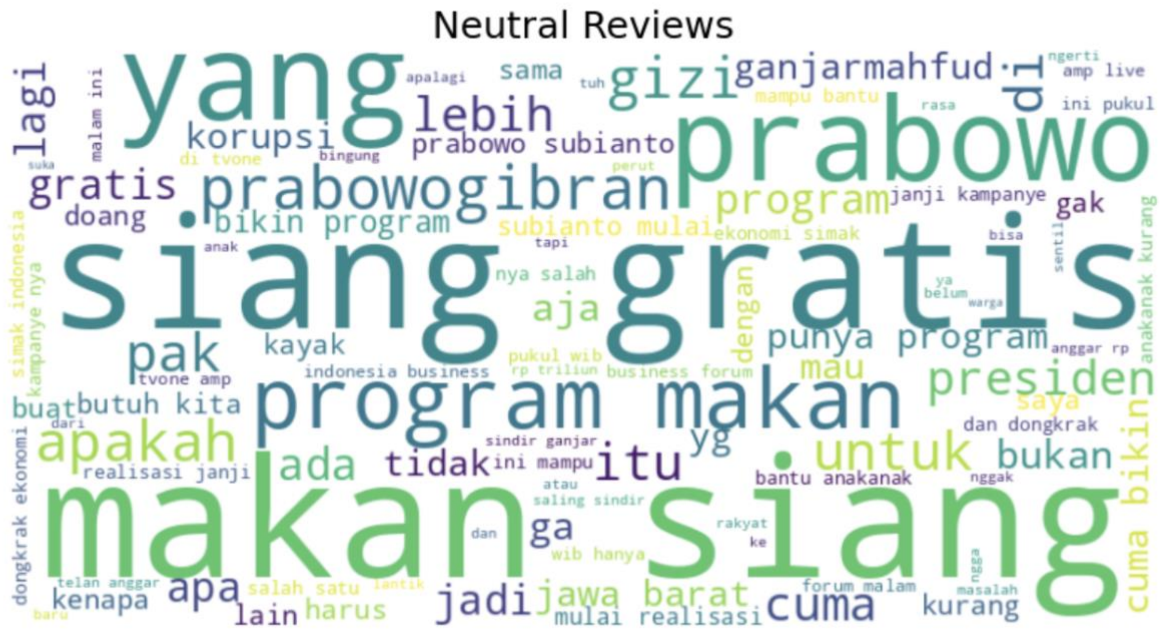


Figure 4.9 World Cloud of Neutral Sentiment

4.2.1 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.

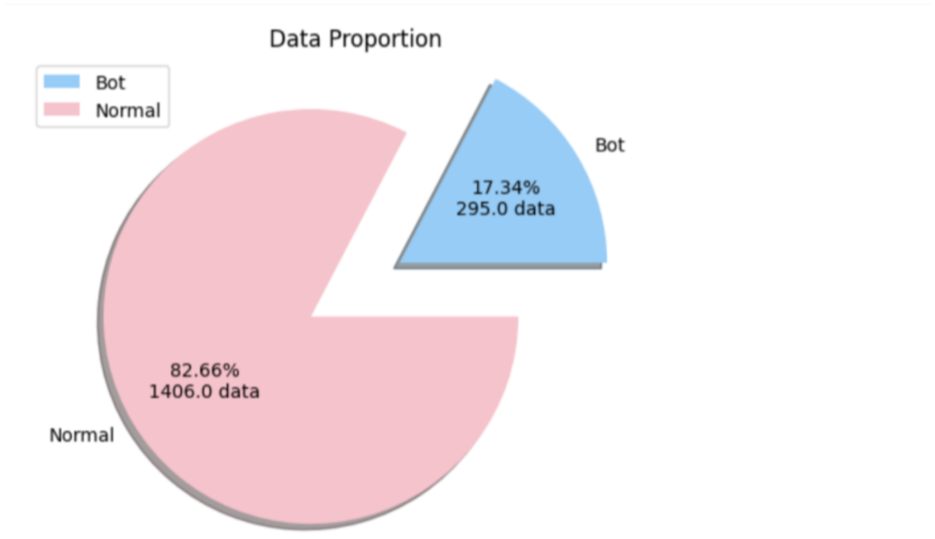


Figure 4.10 Data Proportion

From the graph 4.7 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 17.34% of the total dataset or around 295 data, while the Normal category is the majority of the data, which is around 82.66% or 1406 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 17.34%, which is still considered reasonable to overcome.

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.

4.2.2 Data Cleaning and Preparation

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:

1. **Initialize Sastrawi Stemmer**

   Sastrawi is used to perform stemming, which is changing affixed words into basic forms (root words). For example, "makannya" becomes "makan". Using Stemming helps simplify word variations so that the model can more easily recognize patterns in the data.

2. **Convert Text to Lowercase**

   All letters in the text are converted to lowercase. Makes the analysis more consistent because uppercase and lowercase are treated the same. For example, "PRABOWO" and "prabowo" are considered the same.

3. **Remove URLs**

   Removes links (URLs) from text such as "https://...". Links do not provide relevant sentiment information and can interfere with analysis.

4. **Remove @username**

   Removes mentions or tags such as "@user". Mentions are usually not relevant for sentiment analysis because they only point to a specific account.

5. **Remove Hashtags**

   Removing hashtags such as "#prabowo" or "#makangratis". Hashtags can be removed because they often do not contain the context needed in sentiment analysis, although there are certain cases where hashtags are analyzed separately.

6. **Remove Numbers**

   Removes numbers from text. Numbers usually have no meaning in the context of sentiment, unless specifically relevant (can be processed separately if important).

7. **Remove Punctuation**

   Removes punctuation such as ".", ",", "?", etc. Punctuation does not contribute directly to sentiment analysis.

8. **Remove Extra Whitespace**

   Removes excess whitespace in text. Makes text neater and easier to read.

9. **Apply Stemming**

   Uses the Sastrawi stemmer to convert words to their basic form. Reduces variations in words that have the same meaning.

10. **Apply Preprocessing**

    Combines all the above steps into one preprocessing pipeline that is applied to the entire dataset. Ensures all data is processed in a uniform manner.

11. **Translate Data to Minimize English Words**

Translates English words to Indonesian using a library such as the Google Translate API. Standardizes the language so that all text is in one language (Bahasa Indonesia) to facilitate sentiment analysis.

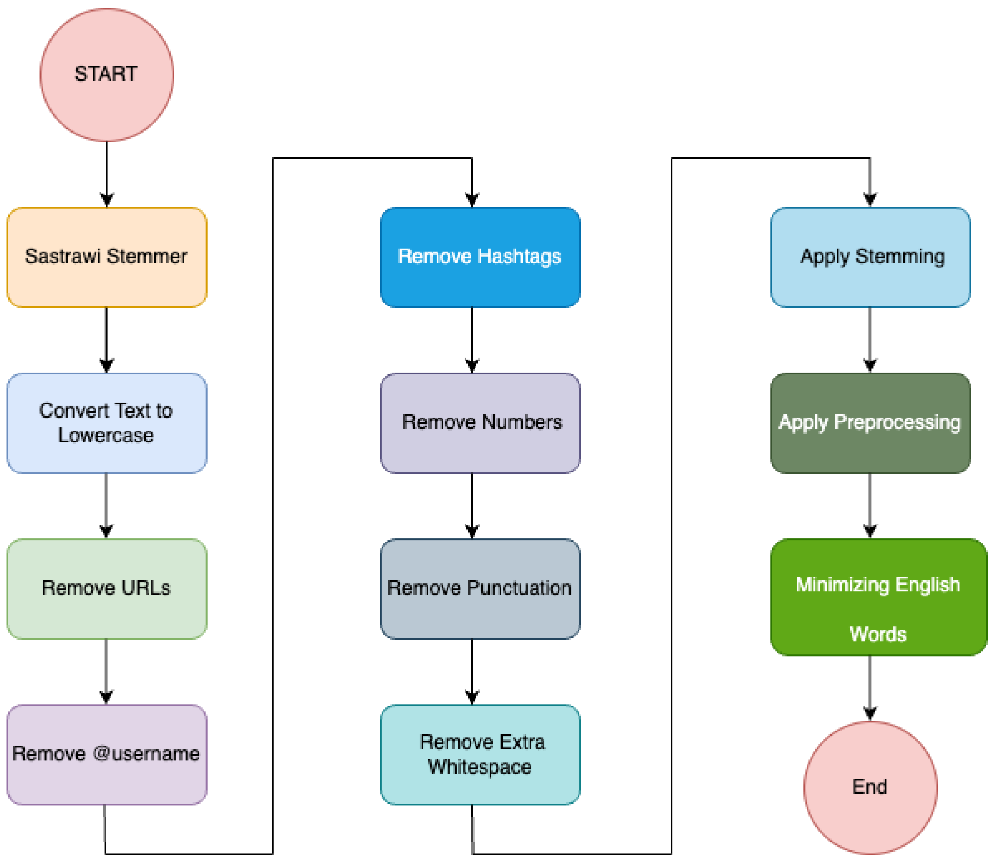


Figure 4.11 Flow Data Cleaning and Preparation

In the image above, it explains the flow of the data cleaning process with a literary stemmer to the process of minimizing words in English.

4.2.3 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.

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

Table 4.2 Some Examples of Sentiment Analysis sentences

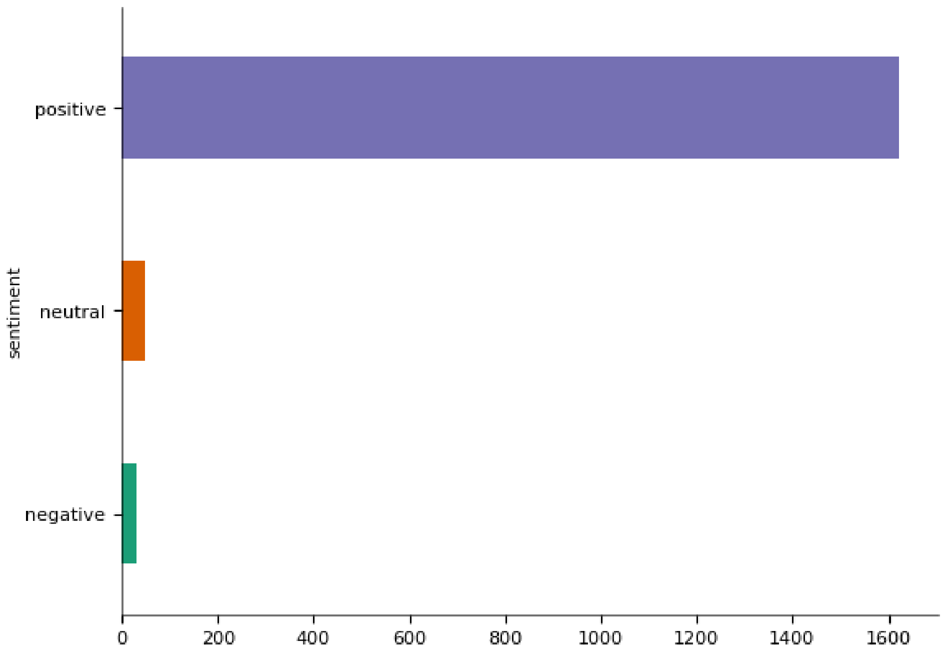


Figure 4.12 Distribution of sentiment analysis result categories

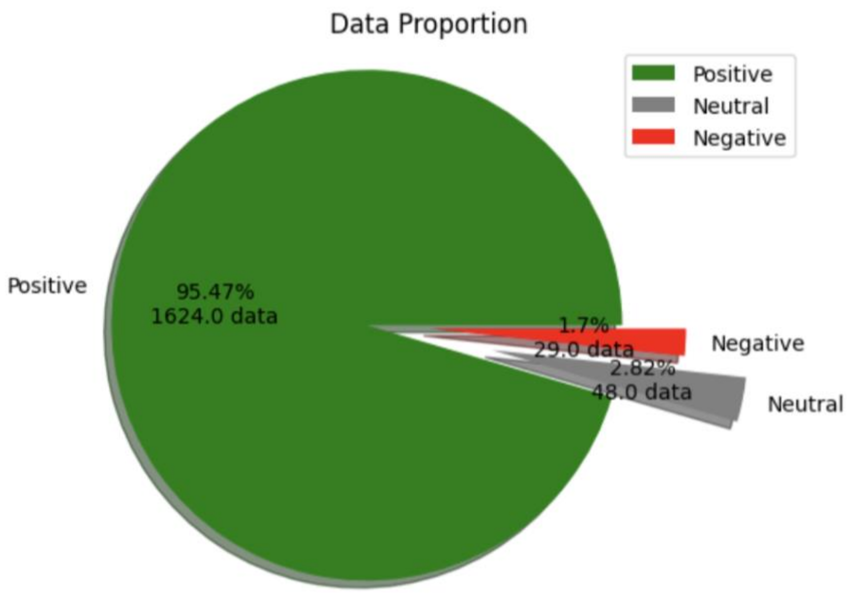


Figure 4.13 Graph of Proportion of data from sentiment analysis results

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.

4.3 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.

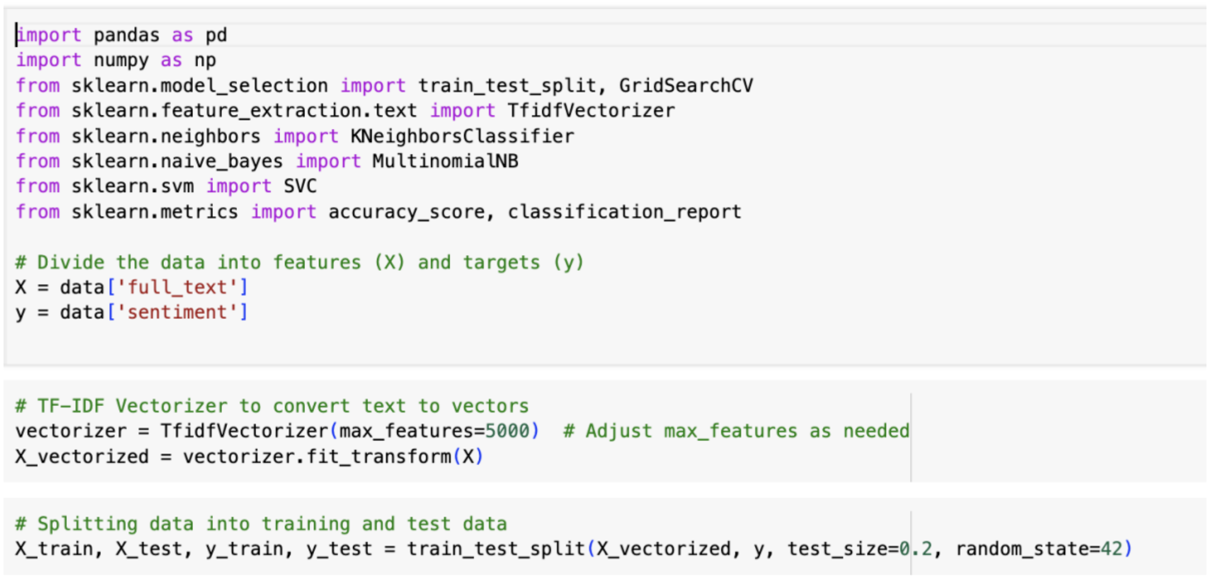
The Importance of TF-IDF in Machine Learning :

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.



```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

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

4.4 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

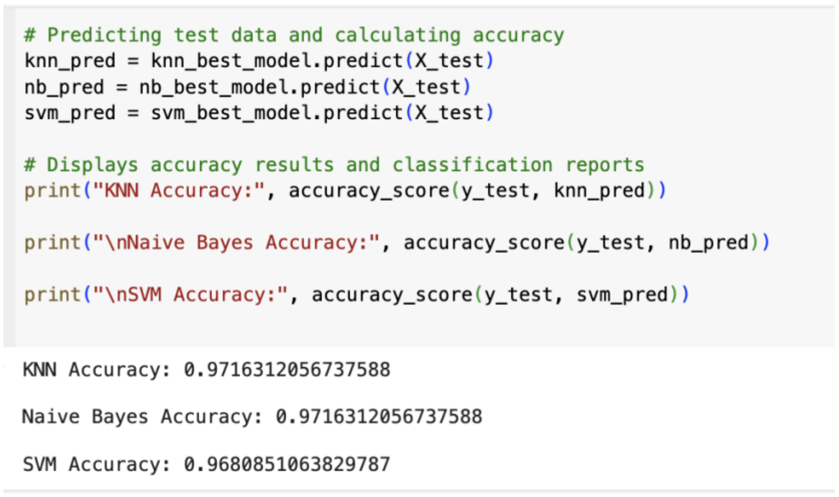
   Accuracy is calculated with the accuracy_score function, which compares the predictions to the original labels in the test data.

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

KNN: 97.61%

Naive Bayes: 97.61%

SVM: 96.80%



```
# 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: 0.9716312056737588

Naive Bayes Accuracy: 0.9716312056737588

SVM Accuracy: 0.9680851063829787
```

Figure 4.15 Accuracy Results and Classification Reports

4.5 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 Naive Bayes have the same high accuracy (97.61%) and are better than SVM (96.80%) 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 95%, which means that the data has been processed well (for example: through TF-IDF Vectorizer and balanced training-test data division). Although SVM 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 Naive Bayes can be selected as the best model for sentiment analysis cases based on performance on test data.