

BAB 4

HASIL DAN PEMBAHASAN

4.1 Implementasi *Pseudocode*

Pada bagian ini akan dijelaskan *pseudocode* yang digunakan dalam membuat model analisis sentimen masyarakat terhadap *childfree* pada *twitter* menggunakan perbandingan metode *random forest* dan *naïve bayes*.

4.1.1 *Pseudocode Scrapping Data*

Dalam proses tahapan *scrapping data*, dimana tahapan ini langkah awal untuk mengambil data dari *Twitter* sebagai *dataset* dan akan diolah ke tahap *preprocessing*. Berikut *pseudocode* tahapan *scrapping data*.

```
import time
from selenium.webdriver import Chrome
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import
WebDriverWait
from selenium.webdriver.support
import expected_conditions as EC
import pandas as pd
import numpy as np
data = pd.DataFrame()
twitter_link = ['url']
twitter_range = [6, 30]
tweetssss = []
list_tweet = []
for tw in range(len(twitter_link)):
    with Chrome(executable_path=r'chromedriver') as
driver:
```

```

print('=====')
print('Processing link', twitter_link[tw])
driver.get(twitter_link[tw])

for i in range(twitter_range[tw]):
    print('Scroll Iteration', i+1)
    #         driver.execute_script('window.scrollTo(0,
document.body.scrollHeight)') # scroll
    driver.execute_script('window.scrollTo(0, 800)') #
scroll
    time.sleep(10) #let content to load

    tweets = driver.find_elements(By.CSS_SELECTOR,
'[data-testid="cellInnerDiv"]')
    for tweet in tweets:
        tweet_d = tweet.find_elements(By.CLASS_NAME, 'r-
kzbnkw')
        for t in tweet_d:
            try:
                not_now = driver.find_element(By.XPATH,
'//*[@id="layers"]/div[2]/div/div/div/div/div/div[2]/
div[2]/div/div[2]/div/div[2]/div[2]/div[2]')
                not_now.click()
            except Exception as e:
                pass
            spans = t.find_elements(By.TAG_NAME, 'span')
            try:
                name = spans[0].find_element(By.CLASS_NAME, 'r-
qvutc0')
                if name is not None:
                    time_s = t.find_element(By.TAG_NAME, 'time')

```

```

        tweet_s      =      t.find_element(By.CSS_SELECTOR,
'[data-testid="tweetText"']

        if tweet_s.text not in tweetssss:
            tweetssss.append(tweet_s.text)
            list_tweet.append([name.text, tweet_s.text])
        except Exception as e:
            pass
data = pd.DataFrame(list_tweet, columns = ['User',
'Tweet'])
data

```

Gambar 4.1 *Pseudocode Scrapping Data*

4.1.2 *Pseudocode Preprocessing (Tanpa Pos Tagger)*

Dalam proses tahapan *preprocessing*, dimana tahapan *preprocessing* ini merupakan langkah-langkah awal yang dilakukan untuk membersihkan dan mempersiapkan data sebelum diolah atau dianalisis. Berikut *pseudocode* tahapan *preprocessing* tanpa menggunakan *pos tagger*:

1. Dalam melakukan *preprocessing* terdapat beberapa *library* yang digunakan untuk pemrosesan data dengan menggunakan modul 'word_tokenize' yang berguna untuk memecah teks menjadi token-token atau unit-unit kecil, seperti kata-kata. Selanjutnya, impor dilakukan untuk *StemmerFactory* dari pustaka Sastrawi, yang memungkinkan implementasi stemming dalam bahasa Indonesia. Selain itu, pustaka *pandas* (pd) diimpor untuk manajemen dan analisis data tabular, dan modul *re* untuk ekspresi reguler. Tahap selanjutnya adalah mengunduh dataset yang dibutuhkan oleh *nlTK*, seperti *stopwords* (kata-kata umum yang sering diabaikan dalam analisis teks) dan model pemecah kata (*punkt*). Ini memastikan bahwa sumber daya yang diperlukan untuk tokenisasi dan pembersihan data teks tersedia. Berikut *pseudocode import library*, dapat dilihat pada Gambar 4.2.

```

import library

begin
    import library nltk
    import library word_tokenize
    import library StemmerFactory
    import library pandas

```

```

import library re
download corpus stopwords
download corpus punkt
end

```

Gambar 4.2 *Pseudocode Import Library*

2. Selanjutnya melakukan *preprocessing* pada data teks. Proses ini melibatkan beberapa tahapan seperti penghapusan karakter khusus, normalisasi kata, tokenisasi, filtering stopwords, stemming, dan klasifikasi sentiment. *Pseudocode* pembentukan proses *preprocessing*, dapat dilihat pada Gambar 4.3.

```

preprocessing
begin
  import stopwords
  stopwords <- set Indonesian
  additional_stopwords <- pd.read_excel
('corpus/Additional_Stopwords.xlsx')
  new_Stopwords <-
stopword.union(additional_stopwords)

  factory <- StemmerFactory()
  stemmer <- factory_createstemmer()

  positive_data <- pd.read_csv
('corpus/positive.txt')
  negative_data <- pd.read+csv
('corpus/negative.txt')

  data_normalisasi <- pd.read_excel
('corpus/normalisasi.xlsx')
  kata_normalisasi <-
data_normalisasi['kata'].tolist()
  hasil_normalisasi <-
data_normalisasi['Normalisasi'].tolist()

  define remove_emojis:
    emoji <- compilasi pola ekspresi
    emoticons
    symbols & pictographs
    transport & map symbols
    flags (ios)
    chines chart
    dingbots
    return replace the pattern in text into a
string (emoj, '', data)

  define case folding(text):
    text <- str(teks).lower()

```

```

text <- remove URLs
text <- remove usernames
text <- remove the # in#hashtag
text <- remove digits number
text <- remove punctuation
text <- remove Emojis
text <- join (text.split(''))
return text

define tokenizing(text):
    min_len_token <- minimal 2 huruf pada kata
    text <- tokenize kata dan kalimat
    text <- [t for t in text if len(t) >
min_len_token]
    return text

define filtering(token):
    filter_ <- filter token in new stopwords
    return " ".join(filter_)

define stemming(filtered_token):
    words <- filtered_token.split()
    backstr <- []
    for w in words:
        stemmed <- stemming (w)
        array stemmed

    return backstr

define check_positive_negative(kata stemming):
    words <- split kata hasil stemming
    positif <- 0
    negative <- 0
    for word in words:
        positive <- jika positive tambahkan ke
count positive
        negative <- jika negative tambahkan ke
count negative
        jika data tidak positive atau negative
akan di skip
        hasil = ""
end

```

Gambar 4.3 *Pseudocode Preprocessing*

3. Setelah melakukan tahapan preprocessing, langkah selanjutnya adalah membaca dataset yang akan dianalisis. *Pseudocode read dataset* dapat dilihat pada Gambar 4.4.

```
read dataset
begin
  data <- read dataset
  data <- remove unnamed
  data
end
```

Gambar 4.4 *Pseudocode Read Dataset*

4. Kemudian melakukan proses pra-pemrosesan pada setiap *tweet* dalam dataset menggunakan fungsi yang telah dibuat pada tahap sebelumnya (Gambar 3.9). *Pseudocode* hasil *preprocessing* dapat dilihat pada Gambar 4.5.

```
hasil preprocessing
begin
  Dataset <- list()
  list_word <- []
  stem_data <- []

  menghasilkan objek iterasi dari DataFrame():
  print('Preprocessing Index' dengan nilai Index)
  text <- r['tweet']
  _casefolding <- casefolding(text)
  _normalisasi <- normalisasi_kata(_casefolding)
  _tokenizing <- tokenizing(_normalisasi)
  _filtering <- filtering(_tokenizing)
  _stemming <- stemming(_filtering)
  _cek_posneg <-
check_positive_negative(_stemming)

  label = _cek_posneg

  _process <- False
  if disable_duplicate == True
    if _stemming text not in list
      _process <- True
      hasil stemming
    else:
      print ('Skip.!' (Duplicate data)',
end')
  else:
    _process <- True

  if _process == True:
```

```

list stemming
Dataset({
  "username"    <- r['username'],
  "tweet"       <- text,
  "casefolding" <- _casefolding,
  "tokenizing"  <- _tokenizing,
  "filtering"   <- _filtering,
  "stem"        <- _stemming,
  "label"       <- label
})
print('Done.!!')
end

```

Gambar 4.5 *Pseudocode Hasil Preprocessing*

- Langkah berikutnya membuat DataFrame dari *preprocessing tweet* yang disimpan dalam variabel 'dataset'. DataFrame ini berisikan informasi yang diproses untuk setiap *tweet*, termasuk teks yang sudah di case-fold, normalisasi, tokenisasi, filtering, dan sebagainya. *Pseudocode* menampilkan dataset berupa DataFrame dapat dilihat pada Gambar 4.6.

```

tampil hasil dataset
begin
  df_dataset <- pd.DataFrame(Dataset)
  df_dataset
end tampil hasil dataset
begin
  df_dataset <- pd.DataFrame(Dataset)
  df_dataset
end

```

Gambar 4.6 *Pseudocode Tampil Hasil Dataset*

- Selanjutnya melakukan pengelompokkan data dalam DataFrame dengan kode program 'groupby', selain itu kode program 'df_dataset' ini berdasarkan nilai pada kolom 'label' dan kemudian menghitung jumlah data pada setiap kelompok. *Pseudocode* pengelompokkan data dapat dilihat pada Gambar 4.7.

```

groupby dataset
begin
  groupby dataset (['label']).size()
end

```

Gambar 4.7 *Pseudocode Groupby Dataset*

2. Selanjutnya melakukan perhitungan frekuensi kemunculan setiap kata dalam 'list_word'. Lalu dari hasil perhitungan frekuensi kemunculan kata-kata dalam 'list_word' dengan membuat DataFrame baru untuk menyajikan kata-kata. *Pseudocode* perhitungan kata dalam list_word dapat dilihat pada Gambar 4.8.

```
list word

begin
  t_list <- []
  for t in list_word:
    for nt in t:
      t_list.append(nt)
  df_t_list <- pd.DataFrame(t_list,
columns=['Word'])
  word_data <- df_t_list.groupby(['Word']).size()

  df_list_words <- pd.DataFrame(word_data, total,
ascending <- False)
  df_list_words
end
```

Gambar 4.8 *Pseudocode List_word*

4.1.3 *Pseudocode Preprocessing (Pos Tagger)*

Dalam penelitian ini terdapat proses *preprocessing* menggunakan *pos tagger*. Berikut *pseudocode* tahapan *preprocessing* sebagai berikut:

1. Tahapan pertama melakukan install terlebih dahulu nlp-id, seperti *pseudocode* dapat dilihat pada Gambar 4.9.

```
install NLP-id

begin
  install nip-id
end
```

Gambar 4.9 *Pseudocode Install NLP-id*

2. Kemudian, melakukan *import* beberapa *library* yang digunakan untuk pemrosesan data dengan menggunakan modul 'word_tokenize' yang berguna untuk memecah teks menjadi token-token atau unit-unit kecil, seperti kata-kata. Selanjutnya, impor dilakukan untuk *StemmerFactory* dari pustaka Sastrawi, yang memungkinkan implementasi stemming dalam bahasa Indonesia. Selain itu, pustaka pandas (pd) diimpor untuk manajemen dan analisis data tabular, dan modul re untuk ekspresi reguler. Tahap selanjutnya adalah mengunduh dataset yang dibutuhkan oleh nltk, seperti stopwords (kata-kata umum yang sering diabaikan dalam analisis teks) dan model

pemecah kata (punct). Ini memastikan bahwa sumber daya yang diperlukan untuk tokenisasi dan pembersihan data teks tersedia. *Pseudocode import library* dapat dilihat pada Gambar 4.10.

```
import library

begin
  import library nltk
  import library word_tokenize
  import library StemmerFactory
  import library pandas as pd
  import library re
  import library RegexpParser
  import library PostTag
  inialisasi postagger <- PostTag()

  download corpus stopwords
  download corpus punct
end
```

Gambar 4.10 *Pseudocode Impor Library*

3. Kemudian, langkah selanjutnya adalah membaca dataset yang akan dianalisis. Berikut *pseudocode read dataset*, dapat dilihat pada Gambar 3.10.
4. Selanjutnya melakukan *preprocessing* pada data teks. Proses ini melibatkan beberapa tahapan seperti penghapusan karakter khusus, normalisasi kata, tokenisasi, filtering stopwords, stemming, dan klasifikasi sentiment. Berikut *pseudocode proses preprocessing*, dapat dilihat pada Gambar 4.11.

```
preprocessing

begin
  import stopwords
  stopwords <- set Indonesian
  additional_stopwords <- pd.read_excel
('corpus/Additional_Stopwords.xlsx')
  new_Stopwords <-
stopword.union(additional_stopwords)

  factory <- StemmerFactory()
  stemmer <- factory_createstemmer()

  positive_data <- pd.read_csv
('corpus/positive.txt')
  negative_data <- pd.read+csv
('corpus/negative.txt')

  data_normalisasi <- pd.read_excel
('corpus/normalisasi.xlsx')
```

```

    kata_normalisasi <-
data_normalisasi['kata'].tolist()
    hasil_normalisasi <-
data_normalisasi['Normalisasi'].tolist()

    define remove_emojis:
        emoji <- compilasi pola ekspresi
            emoticons
            symbols & pictographs
            transport & map symbols
            flags (ios)
            chines chart
            dingbots
        return replace the pattern in text into
a string (emoji, '', data)

    define case_folding(text):
        text <- str(teks).lower()
        text <- remove URLs
        text <- remove usernames
        text <- remove the # in#hastag
        text <- remove digits number
        text <- remove punctuation
        text <- remove Emojis
        text <- join (text.split(' '))
        return text

    define tokenizing(text):
        min_len_token <- minimal 2 huruf pada kata
        text <- tokenize kata dan kalimat
        text <- [t for t in text if len(t) >
min_len_token]
        return text

    define filtering(token):
        filter_ <- filter token in new stopwords
        return " ".join(filter_)

    define stemming(filtered_token):
        words <- filtered_token.split()
        backstr <- []
        for w in words:
            stemmed <- stemming (w)
            array stemmed

        return backstr

    define check_positive_negative(kata
stemming):
        words <- split kata hasil stemming
        positif <- 0
        negatif <- 0
        for word in words:

```

```

        positive <- jika positive tambahkan
ke count positive
        negative <- jika negative tambahkan
ke count negative
        jika data tidak positive atau
negative akan di skip
            hasil = ""

            if (positive > negative):
                hasil = Positive
            elif (positive == negative):
                hasil = Neutral #Jika hasil
positive lebih banyak dari hasil negative
            else:
                hasil = Negative #Jika hasil
negative lebih banyak dari hasil positive
            return hasil

        define pos_tagger(words):
            pos_tag <- fungsi
posttager.get_pos_tag(words)

            return pos_tag

        define extract_data(post_tag):
            grammar <- r""""
N1: {<FW|ADV|NEG|NN>}
""""
            cp <- fungsi parser potongan kata
            result_test <- cp.parse(post_tag)
            word <- []
            membentuk pohon potangan kata
                if(potongan kata dan label () !=
'S'):
                    if subtree.leaves:
                        wrd <- ' '.join([n[0] for n
in subtree.leaves()])
                        word.append(wrd)
            return ' '.join(word)

        end

```

Gambar 4.11 *Pseudocode Preprocessing*

5. Kemudian melakukan implementasi pra-pemrosesan pada setiap *tweet* dalam dataset menggunakan model *preprocessing* yang telah dibuat sebelumnya (Gambar 3.17). *Pseudocode* hasil *preprocessing* dapat dilihat pada Gambar 4.12.

```

hasil preprocessing

begin
  disable_duplicate <- True

  Dataset <- list()
  list_word <- []
  stem_data <- []

  menghasilkan objek iterasi dari DataFrame():
  print('Preprocessing Index' dengan nilai
Index)
  text <- r['tweet']
  _casefolding <- casefolding(text)
  _normalisasi <-
normalisasi_kata(_casefolding)
  _tokenizing <- tokenizing(_normalisasi)
  _filtering <- filtering(_tokenizing)
  _stemming <- stemming(_filtering)
  _pos_tagger <- pos_tagger(_stemming)
  _extract <- extract_data(_pos_tagger)
  _cek_posneg <-
check_positive_negative(_extract)

  label <- _cek_posneg

  _process <- False
  if disable_duplicate == True:
    if _stemming not in stem data
      _process <- True
      hasil stemming
    else:
      print('Skip.! (Duplicate data)',
end=' ')
  else:
    _process <- True

  if _process == True:
    list stemming
    Tampil Dataset({
      "username" <- r['username'],
      "tweet" <- text,
      "casefolding" <- _casefolding,
      "tokenizing" <- _tokenizing,
      "filtering" <- _filtering,
      "stem" <- _stemming,

```

```

        "pos_tagger" <- _pos_tagger,
        "extract_data" <- _extract,
        "label" <- label
    })
    print('Done.!!')
end

```

Gambar 4.12 *Pseudocode* Hasil *Preprocessing* *Pos Tagger*

6. Langkah berikutnya membuat *DataFrame* dari *preprocessing tweet* yang disimpan dalam variabel 'dataset'. *DataFrame* ini berisikan informasi yang diproses untuk setiap *tweet*, termasuk teks yang sudah di case-fold, normalisasi, tokenisasi, filtering, dan sebagainya. *Pseudocde* menampilkan dataset berupa *DataFrame* dapat dilihat pada Gambar 4.6.
7. Selanjutnya melakukan pengelompokkan data dalam *DataFrame* dengan kode program 'groupby', selain itu kode program 'df_dataset' ini berdasarkan nilai pada kolom 'label' dan kemudian menghitung jumlah data pada setiap kelompok. *Pseudocde* pengelompokkan data dapat dilihat pada Gambar 4.7.
8. Selanjutnya melakukan perhitungan frekuensi kemunculan setiap kata dalam 'list_word'. Lalu dari hasil perhitungan frekuensi kemunculan kata-kata dalam 'list_word' dengan membuat *DataFrame* baru untuk menyajikan kata-kata. *Pseudocde* perhitungan kata dalam list_word dapat dilihat pada Gambar 4.8.

4.1.4 *Pseudocode* Model *Naïve Bayes* Dan *Random Forest*

Dalam membentuk model *random forest* dan *naïve bayes* tanpa menggunakan *pos tagger* dan menggunakan *pos tagger* dibuat dalam *pseudocode* yang sama, hanya saja yang membedakan ketika melakukan *load* hasil *preprocessing*. Jika ingin melakukan pemodelan klasifikasi *naïve bayes* dan *random forest* pada hasil *preprocessing* tanpa *pos tagger* dapat menggunakan file hasil *preprocessing* tanpa *pos tagger* dan sebaliknya. Berikut merupakan *pseudocode* yang digunakan dalam membentuk model *naïve bayes* dan *random forest*.

1. *Dataset*

Dataset yang digunakan dalam penelitian ini terbagi menjadi data sekunder dan primer. Data primer dalam penelitian ini berupa pengambilan data dari *Twitter* yang berisi *tweet* opini masyarakat terkait *childfree*. Data yang digunakan dalam penelitian ini berjumlah 1309 *tweet* yang diambil dari tanggal 26 Februari 2023 hingga 08 Maret 2023. Sedangkan data sekunder pada penelitian ini berupa studi literatur terhadap jurnal-jurnal penelitian sebelumnya yang terkait dengan penelitian ini. Dalam melakukan proses analisis sentimen masyarakat dilakukan *import library* yang digunakan,

seperti *numpy*, *pandas*, *seaborn*, *matplotlib.pyplot*, *sklearn.naive bayes*, dan lainnya, yang ditampilkan pada Gambar 4.13.

```
import library model

begin
    import library numpy
    import library pandas
    import library seaborn
    import library matplotlib.pyplot
    import library ComplementNB
    import library RandomForestClassifier
    import library train_test_split
    import library TfidfVectorizer, CountVectorizer
    import library accuracy_score,
confusion_matrix, precision_score, recall_score,
f1_score
end
```

Gambar 4.13 *Pseudocode Import Library*

Dari Gambar 4.13 masing-masing *library* memiliki fungsinya, berikut penjelasan *pseudocode*:

- a. *numpy* (np): Pustaka untuk operasi numerik yang efisien dalam Python. Biasanya digunakan untuk manipulasi data numerik, seperti array dan matriks.
- b. *pandas* (pd): Pustaka untuk manipulasi dan analisis data. *Pandas* menyediakan struktur data yang mudah digunakan seperti *DataFrame*, yang memungkinkan Anda bekerja dengan data tabular.
- c. *seaborn* (sns): *Seaborn* adalah pustaka untuk membuat visualisasi data statistik menggunakan *Matplotlib*. Ini sering digunakan untuk membuat grafik yang lebih menarik dan informatif.
- d. *matplotlib.pyplot* (plt): Modul dari *Matplotlib* yang digunakan untuk membuat visualisasi data, seperti grafik dan plot.
- e. *sklearn.naive_bayes.ComplementNB*: Implementasi algoritma Naive Bayes yang disebut *Complement Naive Bayes* (CNB) dari *scikit-learn*. Algoritma ini sering digunakan untuk klasifikasi teks.
- f. *sklearn.ensemble.RandomForestClassifier*: Implementasi dari algoritma *Random Forest* untuk klasifikasi. *Random Forest* adalah metode ensemble yang membangun beberapa pohon keputusan dan menggabungkan hasil yang diperoleh.
- g. *sklearn.model_selection.train_test_split*: Fungsi untuk membagi dataset menjadi subset pelatihan dan pengujian. Ini penting dalam evaluasi kinerja model.

- h. `sklearn.feature_extraction.text.TfidfVectorizer`, `CountVectorizer`: Alat untuk mengubah teks menjadi representasi numerik yang dapat digunakan oleh model pembelajaran mesin. TF-IDF (Term Frequency-Inverse Document Frequency) dan `CountVectorizer` adalah dua metode umum.
- i. `sklearn.metrics`: Modul yang menyediakan berbagai metrik evaluasi kinerja model, seperti akurasi, matriks kebingungan, presisi, recall, dan nilai F1.

2. *Load Preprocessing Data*

"Load Preprocessing Data" adalah tahap awal dalam analisis data dan pembelajaran mesin di mana data mentah diambil dari sumbernya, seperti file CSV atau basis data. Selanjutnya, data ini mengalami serangkaian langkah pra-pemrosesan untuk memastikan kualitas dan ketergantungan model yang optimal. Proses ini melibatkan tugas seperti menangani nilai-nilai yang hilang, mengkodekan variabel kategori, normalisasi skala data, dan pembagian data menjadi subset pelatihan dan pengujian. *Pseudocode load preprocessing data* dapat dilihat pada Gambar 4.14.

```
load hasil preprocessing
begin
  data <- read hasil_preprocessing.csv
  data
end
```

Gambar 4.14 *Pseudocode Load Preprocessing Data*

Dari Gambar 4.14 terdapat fungsi *Load Preprocessing Data*, berikut penjelasan *pseudocode* tersebut:

- a. `data <- read hasil_preprocessing.csv` digunakan untuk menampilkan hasil *preprocessing*.
- b. `data` digunakan untuk menampilkan dataset yang telah dimuat. Dengan memanggil variabel `data`, kita dapat melihat struktur dan konten dataset, yang memungkinkan kita memeriksa apakah proses muat dan pra-pemrosesan data berjalan dengan benar.

Selanjutnya melakukan pengecekan data pada dataset, apakah ada data yang Null atau tidak pada data tersebut. Berikut *pseudocode* `Is.null` dapat dilihat pada Gambar 4.15.

```

sum data null
  begin
    data null sum ()
  end

```

Gambar 4.15 *Pseudocode* Is.null

Kemudian langkah selanjutnya melihat berapa jumlah data untuk masing-masing label yang dimiliki dapat dilihat pada Gambar 4.16.

```

count label
  begin
    data label counts ()
  end

```

Gambar 4.16 *Pseudocode* Label

3. *Setup X&y*

Langkah berikutnya setup x dan y ini digunakan untuk untuk menentukan nilai x dan y nya, nilai x menggunakan data stem/stemming dan nilai y menggunakan data label, *pseudocode* setup X&y dapat dilihat pada Gambar 4.17.

```

setup X & Y
  begin
    X <- list()
    y <- list()
    for objek iterasi dari DataFrame():
      X menggunakan stem
      y menggunakan label
  end

```

Gambar 4.17 *Pseudocode* Setup X& y

4. *Splitting Data*

Selanjutnya dilakukan split data dengan perbandingan 80% untuk data *training* dan 20% untuk data *testing*. *Pseudocode* split data sebagai berikut, dapat lihat pada Gambar 4.18.


```
split data
begin
  X_test, y_train, y_test <-
train_test_split(x,y, test_size <-0.2, random_state <-
0)
end
```

Gambar 4.18 *Pseudocode Splitting Data*

Berdasarkan Gambar 4.18 digunakan untuk membagi dataset menjadi subset pelatihan dan pengujian. Jika dijabarkan untuk setiap fungsinya, berikut penjelasannya:

- a. X dan y: Variabel X dan y mewakili masing-masing fitur (features) dan label (target) dari dataset. Fitur (X) adalah input yang akan digunakan oleh model untuk membuat prediksi, sedangkan label (y) adalah nilai yang ingin diprediksi.
- b. train_test_split: Ini adalah fungsi dari scikit-learn yang digunakan untuk membagi dataset menjadi subset pelatihan dan pengujian. Parameter utama dari fungsi ini adalah X dan y, yaitu fitur dan label dataset.
- c. test_size=0.2: Parameter ini menentukan proporsi dataset yang akan dialokasikan untuk pengujian. Dalam contoh ini, 20% dari data akan digunakan sebagai data pengujian, sedangkan 80% akan menjadi data pelatihan.
- d. random_state=0: Parameter ini menyediakan nilai seed untuk pengacakan (randomization). Ini memastikan bahwa pembagian dataset menjadi pelatihan dan pengujian akan tetap konsisten setiap kali kode dijalankan. Penggunaan nilai yang sama untuk random_state memungkinkan hasil yang dapat direproduksi.
- e. X_train, X_test, y_train, y_test: Variabel ini adalah hasil dari pemanggilan fungsi train_test_split. X_train dan y_train adalah subset pelatihan yang berisi fitur dan label untuk melatih model, sedangkan X_test dan y_test adalah subset pengujian yang akan digunakan untuk menguji kinerja model.

Berikut visualisasi *training* data yang digunakan untuk membuat visualisasi dan menyimpan data pelatihan (*train* data) ke dalam file CSV. *Pseudocode* visualisasi *train* data dapat dilihat pada Gambar 4.19.

```

visualisasi train data

begin
  train_data <- tampil stem dan label dalam
list()
  for setiap element dalam X_tran menggunakan
indeks xt
    add each element X_train[xt] dalam stem
    add each element y_train[xt] dalam label
  conversion train_data <-
pd.DataFrame(train_data)
end

```

Gambar 4.19 *Pseudocode Training Data*

Selanjutnya terdapat *pseudocode* yang digunakan untuk membuat visualisasi dan menyimpan data pengujian (*test data*) ke dalam file CSV. Berikut visualisasi *testing data*, dapat dilihat pada Gambar 4.20.

```

visualisasi test data

begin
  train_data <- tampil stem dan label dalam list()
  for setiap element dalam X_tran menggunakan indeks
xt
  add each element X_test[xt] dalam stem
  add each element y_test[xt] dalam label
  conversion test_data <- pd.DataFrame(test_data)
end

```

Gambar 4.20 *Pseudocode Testing Data*

5. *Delta TF-IDF*

Delta TF-IDF adalah variasi dari metode pengukuran bobot dalam analisis teks yang memodifikasi skema TF-IDF tradisional dengan memasukkan nilai delta. Dalam konteks ini, delta mengacu pada perubahan atau perbedaan dalam frekuensi relatif suatu kata antara dua dokumen atau lebih. Berikut untuk *source code* nya dimana kode tersebut menggunakan *library* 'Sklearn_deltatfidf' untuk menerapkan Delta TF-IDF pada data teks, dapat dilihat pada Gambar 4.21.

```

delta TF-IDF

begin
  import library DeltaTfidfVectorizer from
sklearn_deltatfidf

      vectorizer <- pembobotan vektor gunakan fungsi
library DeltaTfidfVectorizer()
X_train_vec <- vektorisasi data (X_train,
y_train) dalam bentuk array
X_test_vec <- vektorisasi (X_test) pada array

      feature_names <- vektorisasi feature names
matrix
end

```

Gambar 4.21 *Pseudocode Delta TF-IDF*

Selanjutnya menampilkan hasil delta TF-IDF train data, berikut untuk *pseudocode nya*, dapat dilihat pada Gambar 4.22.

```

hasil delta TF-IDF data train

begin
  tfidf_train <- tampil DataFrame(vektorisasi
X_train, index <- loop D{i+1} in range (jumlah karakter
X_train), columns <- feature_names)
  tfidf_train['label'] <- y_train
  simpan hasil tfidf_train.to_csv('tfidf/TF-IDF
data train.csv')
  tampil hasil tfidf_train
end

```

Gambar 4.22 *Pseudocode Delta TF-IDF Train Data*

Selanjutnya menampilkan hasil delta TF-IDF testing data, berikut untuk *pseudocode* dapat dilihat pada Gambar 4.23.

```

hasil delta TF-IDF data test

begin
  tfidf_test <- tampil DataFrame(vektorisasi
X_test, index <- loop D{i+1} in range (jumlah karakter
X_test), columns <- feature_names)
  tfidf_train['label'] <- y_test
  simpan hasil tfidf_test.to_csv('tfidf/TF-IDF
data test.csv')
  tampil hasil tfidf_test
end

```

Gambar 4.23 *Pseudocode Delta TF-IDF Train Data*

6. *Naïve Bayes*

Selanjutnya dilakukan *complement naïve bayes* merupakan adaptasi dari *multinomial naïve bayes* yang dirancang untuk menangani dataset yang tidak seimbang. Kemudian CNB biasanya memberikan hasil yang terdapat perbedaan signifikan dalam kelas pada data. *Pseudocode* untuk analisis sentimen dengan menggunakan *complement naïve bayes* dapat dilihat pada Gambar 4.24.

```
complement NB
begin
  clf <- gunakan fungsi ComplementNB() untuk
adaptasi dari Multinomial Naive Bayes (MNB) pada
imbalance dataset
  clf.fit(X_train_vec, y_train)
  pred_nb <- predict (X_test_vec)
end
```

Gambar 4.24 *Pseudocode ComplementNB*

7. *Random Forest*

Setelah melakukan analisis sentimen menggunakan *Naive Bayes*, berikutnya melakukan analisis sentimen menggunakan *Random Forest*. *Pseudocode* analisis sentiment menggunakan *random forest* dapat dilihat pada Gambar 4.25.

```
model random forest
begin
  max_depth_rf      <- save max dept value []
  pred_rf           <- save the random forest
prediction result label []
  acc_rf            <- save accuracy results []
  max_depth_range  <- rang nilai max depth
np.linspace(10, 180, num = 18)

  for i in max_depth_range:
    print('Proses max dept', int(i))
    add the max depth value to the max_depth rf
variable max_depth_rf.append(i)

    rf <- implementasi
RandomForestClassifier(random_state=1,
max_depth=int(i))
    training model rf.fit dengan X_train &
y_train

    pred <- rf.predict(X_test_vec)
```

```

pred_rf      Add prediction result labels to the list
pred_rf      nilai acc_rf.append(accuracy_score(y_test,
pred))      print('Done...')
pred_rf      end

```

Gambar 4.25 *Pseudocode Random Forest*

4.1.5 *Pseudocode Evaluasi Metode Naïve Bayes Dan Random Forest*

Evaluasi dilakukan menggunakan *confusion matrix* untuk mengetahui performa model *naïve bayes* dan *random forest* tanpa *pos tagger* dan menggunakan *pos tagger*. Berikut akan dijelaskan *pseudocode* untuk evaluasi model klasifikasi metode *naïve bayes* dan *random forest*.

1. Metode *Naïve Bayes*

Dalam penelitian ini membuat dan menampilkan *confusion matrix* serta heatmap dari Hasil prediksi model *complement naive bayes* menggunakan *pseudocode* yang dapat dilihat pada Gambar 4.26.

```

heatmap confusion matrix

begin
    cm_nb      <- confusion_matrix dengan
membandingkan hasil pred_nb dengan y_test, jumlah label
= ['Positive', 'Neutral', 'Negative']
    class_label <- Positive, Neutral, Negative
    df_confusion <- pd.DataFrame(cm_nb, index <-
class_label, columns <- class_label)

    membuat heatmap(df_confusion, annot=True, fmt =
"d", cmap=plt.cm.Blues)
    plot title('Confusion Matrix')
    plot xlabel('Prediction Label')
    plot ylabel('Actual Label')
    plot show()
end

```

Gambar 4.26 *Pseudocode Heatmap Confusion Matrix*

Berikut *pseudocode confusion matrix* dapat dilihat pada Gambar 4.27.

```
nilai confusion matrix

begin
  inisialisasi _accuracy <- bulatkan (hasil
accuracy_score dari nilai(y_test, pred_nb)*100, 2)
  inisialisasi _preciision <- bulatkan(hasil
precision_score dari nilai(y_test, pred_nb,
average='weighted')*100, 2)
  inisialisasi _recall <-
bulatkan(recall_score(y_test, pred_nb,
average='weighted')*100, 2)
  inisialisasi _fscore <-
bulatkan(fl_score(y_test, pred_nb,
average='weighted')*100, 2)

  print('Accuracy   :', get nilai _accuracy, '%')
  print('Preciision :', get nilai _preciision,
'%')
  print('Recall     :', get nilai _recall, '%')
  print('F-Score   :', get nilai _fscore, '%')
end
```

Gambar 4.27 *Pseudocode Menampilkan Confusion Matrix*

Selanjutnya melakukan pengujian data dengan menggunakan beberapa label seperti positif, negative, dan netral. Berikut *pseudocode testing data*, dapat dilihat pada Gambar 4.28.

```
confusion matrix pada data test

begin
  test_data['pred_label_nb'] <- hasil prediksi
naive bayes pred_nb
  tampil hasil test_data
end
```

Gambar 4.28 *Pseudocode Testing Data*

2. Metode *Random Forest*

Setelah pembentukan model *random forest* dilakukann visualisasi ke dalam bentuk plot, untuk melihat hasil maksimum akurasi, serta nilai max dept terbaik. Berikut *pseudocode* yang digunakan dapat dilihat Pada Gambar 4.29.

```
visualisasi grafik plot

begin
  plot figure(figsize=(10,6))
```

```

        plot(max_depth_range, accuracy random forest,
color = 'blue', linestyle='dashed', marker='o',
markerfacecolor='red', markersize=10)
        plot title('accuracy vs. Max Depth')
        plot xlabel('Max Depth')
        plot ylabel('Accuracy')
        print("Maximum accuracy:", round(max(acc_rf),
4), "at max depth =",
int(max_depth_range[acc_rf.index(max(acc_rf))]))
        print('Maka nilai max depth terbaik adalah :',
int(max_depth_range[acc_rf.index(max(acc_rf))]))
    end
end

```

Gambar 4.29 Pseudocode Visualisasi Grafik Plot

Setelah melakukan visualisasi berupa plot, berikutnya melakukan evaluasi menggunakan *confusion matrix*, dengan tujuan mengetahui performas metode *random forest* dalam analisis sentimen masyarakat terhadap *chidfree* di *Twitter*. Berikut untuk *pseudocode*, dapat dilihat Pada Gambar 4.30.

```

confusion matrix random forest

    begin
        inialisasi 85variable best_prediction_rf <-
pred_rf[acc_rf.index(max(acc_rf))]

        inialisasi 85variable cm_rf <-
confusion_matrix(hasil_y_test, best_prediction_rf,
labels=['Positive', 'Neutral', 'Negative'])
        class_label <- ['Positive', 'Neutral',
'Negative']
        df_confusion <- tampil DataFrame (nilai
confusion matrix random forest, index <- class label,
columns = class_label)

        call function sns.heatmap(df_confusion,
annot=True, fmt = "d", cmap=plt.cm.Blues)
        plot title('Confusion Matrix')
        plot xlabel('Prediction Label')
        plot ylabel('Actual Label')
        plot show()
    end
end

```

Gambar 4.30 Pseudocode Confusion Matrix

Berikut *pseudocode confusion matrix* dapat dilihat pada Gambar 4.31.

```
nilai confusion matrix random forest

begin
  inialisasi _accuracy_rf   <- round(hasil
accuracy_score dari nilai(y_test,
best_prediction_rf)*100, 2)
  inialisasi _preiission_rf <- round(hasil
precision_score dari nilai (y_test, best_prediction_rf,
average='weighted')*100, 2)
  inialisasi _recall_rf     <- round(hasil
recall_score dari nilai (y_test, best_prediction_rf,
average='weighted')*100, 2)
  inialisasi _fscore_rf     <- round(hasil
f1_score dari nilai (y_test, best_prediction_rf,
average='weighted')*100, 2)

  print('Accuracy   :', get nilai _accuracy_rf,
'%')
  print('Preiission :', get nilai _preiission_rf,
'%')
  print('Recall     :', get nilai _recall_rf,
'%')
  print('F-Score   :', get nilai _fscore_rf,
'%')
end
```

Gambar 4.31 *Pseudocode Menampilkan Confusion Matrix*

Selanjutnya melakukan pengujian data dengan menggunakan beberapa label seperti positif, negative, dan netral. Berikut *pseudocode testing data*, dapat dilihat pada Gambar 4.32.

```
confusion matrix random forest pada data test

begin
  inialisasi test_data dengan menggunakan
parameter['pred_label_rf'] <- hasil prediksi
best_prediction_rf
  get test_data.to_excel dengan nama
file('Hasil.xlsx')
  tampil hasil test_data
end
```

Gambar 4.32 *Pseudocode Testing Data*

Setelah melakukan beberapa tahapan, terdapat hasil akhir yang diperoleh dari eksekusi pembuatan dataframe menggunakan *library pandas* untuk menampilkan hasil evaluasi model *naïve bayes* dan *random forest*. Berikut untuk *pseudocode*-nya, dapat dilihat Pada Gambar 4.33.

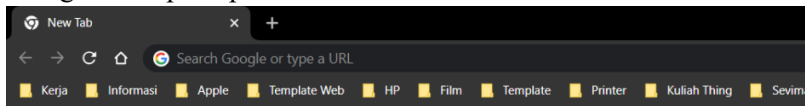
```
hasil evaluasi naive bayes dan random forest

begin
  inisialisasi eval_name <- ['Accuracy',
'Precision', 'Recall', 'F1-score']
  inisialisasi hasil_nb <- [_accuracy,
_precision, _recall, _fscore]
  inisialisasi hasil_rf <- [_accuracy_rf,
_precision_rf, _recall_rf, _fscore_rf]
  inisialisasi hasil <- tampil DataFrame
pd.DataFrame({'': eval_name,
parameter hasil_nb,
'Naive Bayes (%)': call
'Random Forest (%)':
call parameter hasil_rf})
  call hasil
end
```

Gambar 4.33 *Pseudocode Hasil Evaluasi Naïve Bayes dan Random Forest*

4.2 Contoh Desain Antarmuka

Pada Gambar 4.34 dan Gambar 4.35 terlihat desain antar muka awal (*prototype*) yang akan digunakan pada penelitian ini:

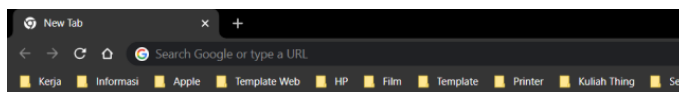


Analisis Data Sentimen

Lakukan import data dari scrapping untuk mengetahui hasil analisis sentimen dengan metode Random Forest dan Naive Bayes



Gambar 4.34 Desain Antar Muka Pertama (*Input*)



Upload Sukses!

Berikut hasil analisa sentimen dan hasil akurasi dari dua algoritma *Random Forest* dan *Naive Bayes*

Analisa Sentimen

No.	Sentimen	Positif	Netral	Negatif
1	Kenapa punya anak? Childfree agar awet muda			√
2	Kita childfree, kalau dituruti bisa jadi 18 anak			√
3	Terkadang childfree juga dibutuhkan		√	
4	Aku tidak mau childfree, cukup satu anak aja	√		
5	Kalau sudah menikah, susah childfree			√
...	...			

Tingkat Akurasi (*Confusion Matrix*)

No.	Klasifikasi	Akurasi
1	<i>Random Forest</i>	97,50%
2	<i>Naive Bayes</i>	93,30%



Gambar 4.35 Desain Antar Muka Kedua (*Output*)

4.3 Percobaan Metode *Naïve Bayes* Dan *Random Forest* Tanpa Pos Tagger

Dalam implementasi metode *naïve bayes* dan *random forest* dilakukan menggunakan bahasa pemrograman Python untuk proses klasifikasi pada dataset *childfree* pada Twitter. Dataset penelitian ini berupa pengambilan data dari *Twitter* yang berisi *tweet* opini masyarakat terkait *childfree*. Data yang digunakan dalam penelitian ini berjumlah 1309 *tweet* yang diambil dari tanggal 26 Februari 2023 hingga 08 Maret 2023. Dataset yang telah dikumpulkan dilakukan *load* dataset untuk melihat data tersebut yang hasilnya dapat dilihat pada Tabel 4.1.

Tabel 4.1 *Read Dataset*

	id	name	web	created_at	username	tweet	location
0	1.63338E+18	lha.iki.lho	NaN	2023-03-08 08:14:58+00:00	lha_iki_lho	Childfree?nKalian gak bakal ngerasain nyuruh ...	NaN
1	1.63337E+18	Kore of Chthonia	NaN	2023-03-08 07:43:30+00:00	saturnomeus	Happy Women's Day untuk single mothers, peremp...	NaN
2	1.63337E+18	Habibi Habibah	https://t.co/Yr7t1qvBYJ	2023-03-08 07:20:42+00:00	ibnurozii	@DitjenPajakRI Hufftt mungkin skrng dipikranny...	Depan Layar
3	1.63335E+18	Dorri	NaN	2023-03-08 06:27:52+00:00	ggakenna	@detikcom Sepandai apapun kalian menyimpan ban...	NaN
4	1.63334E+18	-	NaN	2023-03-08 05:47:45+00:00	vnlylp	@Askrfess Zalim bgt ibunya, yang begini begini...	NaN
...
1304	1.62975E+18	Ajeng Kartika N	NaN	2023-02-26 07:49:23+00:00	ajengkkartika	Aku emak emak anak 3 malah ngewanti wanti teme...	Malang, East Java
1305	1.62975E+18	???????? ??????????????????????	https://t.co/jp2sddEIQp	2023-02-26 07:41:56+00:00	ChacaVarma	Kalian terlalu sibuk sama tantrum dan mulut ja...	Upper earth
1306	1.62975E+18	rosaa	https://t.co/HFzP9lvdH	2023-02-26 07:40:56+00:00	psiholic	@zhafiraqyla Sendernya kepo bgt bjir, org baru...	Universitas Gadjah Mada
1307	1.62975E+18	Sapiens?????	NaN	2023-02-26 07:31:24+00:00	Homo_sapiens21	Twit ini kejam karena survival of the fittest ...	NaN
1308	1.62975E+18	Yokanang C. A.	https://t.co/e9dYqtIIVO	2023-02-26 07:31:04+00:00	Yokanang_	@Greessel_JKT48 kucingmu gak childfree cel?	NaN

Datset tersebut dilakukan *preprocessing* yang terdiri dari *case folding*, *tokenizing*, *filtering* dan *stemming* tanpa menggunakan *pos tagger* yang hasilnya dapat dilihat pada Gambar 4.2.

Tabel 4.2 Hasil *Preprocessing*

	username	tweet	casefolding	tokenizing	filtering	stem
0	lha_iki_lho	Childfree?nKalian gak bakal ngerasain	childfree?nkalian gak bakal ngerasain	[childfree, kalian, tidak, bakal,	childfree ngerasain nyur	childfree ngerasain
1	saturnomeus	Happy Women's Day untuk single moth	happy womens day untuk single moth	[happy, womens, day, untuk, si	happy womens day singi	happy womens day
2	ibnurozii	@DitjenPajakRI Hufftt mungkin skrng dip	hufftt mungkin skrng dipikrannya child	[hufftt, mungkin, sekarang, dipik	dipikrannya childfree bu	pikir childfree buruk
3	ggakenna	@detikcom Sepandai apapun kalian me	sepandai apapun kalian menyimpan b	[sepandai, apapun, kalian, men	sepandai apapun menyii	pandai apa simpan
4	vnlylp	@Askrfess Zalim bgt ibunya, yang begin	zalim bgt ibunya yang begini begini ini	[zalim, begitu, ibunya, yang, bej	zalim ibunya childfree	zalim ibu childfree
...
1248	ajengkkartika	Aku emak emak anak 3 malah ngewanti	aku emak emak anak malah ngewanti	[aku, emak, emak, anak, malah	emak emak anak ngewa	emak emak anak ng
1249	ChacaVarma	Kalian terlalu sibuk sama tantrum dan r	kalian terlalu sibuk sama tantrum dan	[kalian, terlalu, sibuk, sama, tar	sibuk tantrum mulut jah	sibuk tantrum mulu
1250	psiholic	@zhafiraqyla Sendernya kepo bgt bjir,	sendernya kepo bgt bjir org baru nikal	[sendernya, kepo, begitu, bjir, c	sendernya kepo bjir ora	sender kepo bjir org

Dari Gambar 4.2 hasil *preprocessing* data, dilakukan pengelompokan jumlah label dari setiap label. Dalam penelitian ini menggunakan 3 jenis label yaitu Negative dengan total data 331, Neutral berjumlah 627 data dan Positive berjumlah 295 data.

Selanjutnya, dilakukan pembagian dataset menjadi 2 jenis data yaitu data *train* 80 % dan data *test* 20 % yang dapat dilihat pada Gambar 4.3 dan Tabel 4.3. Pada data *train* berjumlah 1002 data dan data *test* berjumlah 251 data.

Tabel 4.3 Data *Training*

	stem	label
0	liat bawa childfree	Neutral
1	kampanye childfree dosa rendah anak indah utam...	Negative
2	cocok childfreetanpa kondomanti tanggal merah	Positive
3	ohh alas marak childfree	Neutral
4	childfree prolife tunggu lahir buang	Positive
...
997	endeavor bunda rei childfree husbu gwe	Neutral
998	kucing anut childfree	Positive
999	teh rafael mending mah childfree	Neutral
1000	awet muda childfree	Positive
1001	gilir kayak gin gamau dukung dilakuin ortu bab...	Negative

Tabel 4.4 Data *Testing*

	stem	label
0	childfree gittasav tunggu pensi	Positive
1	duhh bingung pikir childfree gitasav tolol kay...	Negative
2	minimal pikirin childfree	Neutral
3	allah jahat anak someday against orang milih c...	Negative
4	andai gitasav hidup paradisi kampanye childfree	Neutral
...
246	salah gitasav ngomong childfree orang juang be...	Negative
247	ngebahas childfree kadang nyelutuk hah ngapain...	Negative
248	suruh childfree anak tega	Neutral
249	betul berat orgtua pantesan childfree liat sis...	Negative
250	tuju childfree saiki goblok arek enom	Neutral

Pada masing-masing data dilakukan pembobotan kata menggunakan TF-IDF tradisional dengan memasukkan nilai delta. Dalam konteks ini, delta mengacu pada perubahan atau perbedaan dalam frekuensi relatif suatu kata antara dua dokumen atau lebih. yang hasilnya dapat dilihat pada Gambar 4.5 dan Gambar 4.6.

Tabel 4.5 Hasil *Delta TFIDF Train Data*

	aahhma ma	abah	abai	abdillah	abg	abisabis an	abissss	aborsi	about	absurd	...
D1	0	0	0	0	0	0	0	0	0	0	...
D2	0	0	0	0	0	0	0	0	0	0	...
D3	0	0	0	0	0	0	0	0	0	0	...
D4	0	0	0	0	0	0	0	0	0	0	...
D5	0	0	0	0	0	0	0	0	0	0	...
...
D998	0	0	0	0	0	0	0	0	0	0	...
D999	0	0	0	0	0	0	0	0	0	0	...
D1000	0	0	0	0	0	0	0	0	0	0	...
D1001	0	0	0	0	0	0	0	0	0	0	...
D1002	0	0	0	0	0	0	0	0	0	0	...

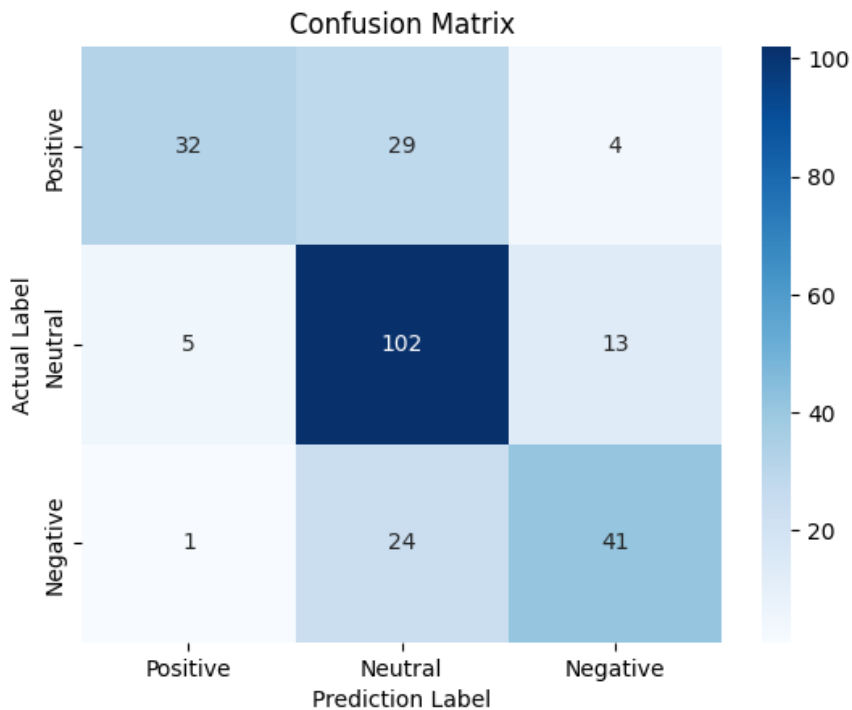
Tabel 4.6 Hasil *Delta TFIDF Testing Data*

	aahhma ma	abah	abai	abdillah	abg	abisabis an	abissss	aborsi	about	absurd	...
D1	0	0	0	0	0	0	0	0	0	0	...
D2	0	0	0	0	0	0	0	0	0	0	...
D3	0	0	0	0	0	0	0	0	0	0	...
D4	0	0	0	0	0	0	0	0	0	0	...
D5	0	0	0	0	0	0	0	0	0	0	...
...
D998	0	0	0	0	0	0	0	0	0	0	...
D999	0	0	0	0	0	0	0	0	0	0	...
D1000	0	0	0	0	0	0	0	0	0	0	...
D1001	0	0	0	0	0	0	0	0	0	0	...
D1002	0	0	0	0	0	0	0	0	0	0	...

Selanjutnya, dilakukan pembentukan model *naïve bayes* dan *random forest* untuk melatih model menggunakan data *train* dan data *test* serta dilakukan evaluasi menggunakan *confusion matrix* untuk mengetahui performa model dalam klasifikasi. Berikut merupakan hasil *confusion matrix* metode *naïve bayes* dan *random forest*.

1. Hasil Evaluasi Metode *Naïve Bayes*

Metode *naïve bayes* dilakukan evaluasi menggunakan *confusion matrix* pada data *test* yang mendapatkan hasil pada Gambar 4.36.



Gambar 4.36 *Confusion Matrix* Metode *Naïve Bayes* Tanpa *Pos Tagger*
Confusion Matrix

Pada Gambar 4.36 dapat dijelaskan bahwa metode *naïve bayes* tanpa *pos tagger* diperoleh hasil prediksi label *Positive* berjumlah 65 data yang diuji, terdapat 32 data yang terklasifikasi dengan benar dan terdapat kesalahan prediksi berjumlah 29 data yang masuk kedalam label *Neutral* dan 4 data masuk kedalam label *Negative*. Sedangkan, pada label *Neutral* dengan jumlah data 120 yang diuji, terdapat 102 data yang terklasifikasi dengan benar, kesalahan prediksi berjumlah 5 data yang masuk kedalam label *Positive* dan 13 data pada label *Negative*. Pada label *Negative* dengan jumlah data 66 data yang diuji, terdapat 41 data yang terklasifikasi dengan benar, terdapat kesalahan prediksi berjumlah 1 data yang masuk kedalam label *Positive* dan 24 data masuk kedalam label *Neutral*.

Tabel 4.7 Hasil *Confusion Matrix* Metode *Naïve Bayes* Tanpa *Pos Tagger*

Accuracy	69.72%
Precision	71.86%
Recall	69.72%
F-Score	68.94%

Pada Tabel 4.7 hasil evaluasi metode *naïve bayes* tanpa menggunakan *pos tagger* mendapatkan nilai *accuracy* 69.72 %, *precision* sebesar 71.86 %, *recall* sebesar 69.72 % dan *F-Score* sebesar 68.94 %.

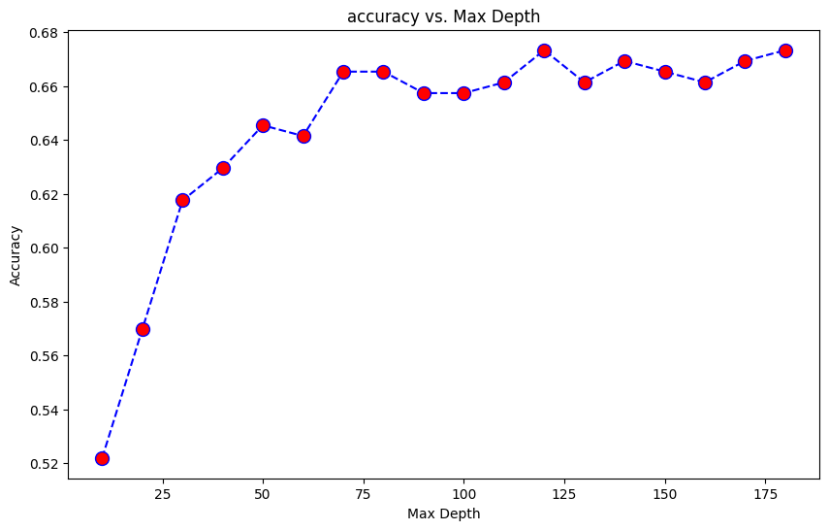
Tabel 4.8 Hasil Klasifikasi *Naïve Bayes* Tanpa *Pos Tagger*

	stem	label	pred_label_nb
0	childfree gittasav tunggu pensi	Positive	Positive
1	duhh bingung pikir childfree gitasav tolol kay...	Negative	Negative
2	minimal pikirin childfree	Neutral	Neutral
3	allah jahat anak someday against orang milih c...	Negative	Negative
4	andai gitasav hidup paradisi kampanye childfree	Neutral	Neutral
...
246	salah gitasav ngomong childfree orang juang be...	Negative	Negative
247	ngebahas childfree kadang nyelutuk hah ngapain...	Negative	Negative
248	suruh childfree anak tega	Neutral	Neutral
249	betul berat orgtua pantasan childfree liat sis...	Negative	Negative
250	tuju childfree saiki goblok arek enom	Neutral	Neutral

Pada Tabel 4.8, merupakan hasil klasifikasi *naïve bayes* tanpa *post tagger* dengan total data 251 data.

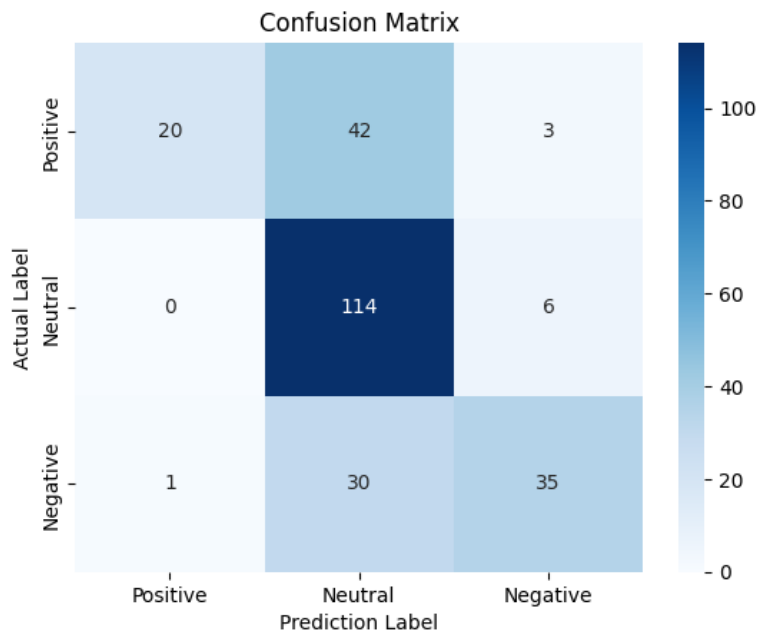
2. Hasil Evaluasi Metode *Random Forest*

Dalam metode *random forest* dilakukan visualisasi plot untuk mendapatkan hasil *maximum accuracy* dan nilai *max dept*. *Maximum accuracy* sebesar 0.6733 at *max depth* = 120 dan nilai *max dept* terbaik adalah 120. Hasil visualisasi plot dapat dilihat pada Gambar 4.37.



Gambar 4.37 Hasil Visualisasi Plot Metode *Random Forest* Tanpa *Pos Tagger*

Kemudian dilakukan evaluasi menggunakan *confusion matrix* yang didapatkan hasil pada Gambar 4.38.



Gambar 4.38 *Confusion Matrix* Metode *Random Forest* Tanpa *Pos Tagger*

Pada Gambar 4.38 dapat dijelaskan bahwa metode *random forest* tanpa *pos tagger* diperoleh hasil prediksi label *Positive* berjumlah 65 data yang diuji, terdapat 20 data yang terklasifikasi dengan benar dan terdapat kesalahan prediksi berjumlah 42 data yang masuk kedalam label *Neutral* dan 3 data masuk kedalam label *Negative*. Sedangkan, pada label *Neutral* dengan jumlah data 120 yang diuji, terdapat 114 data yang terklasifikasi dengan benar, kesalahan prediksi berjumlah 16 data pada label *Negative*. Pada label *Negative* dengan jumlah data 66 data yang diuji, terdapat 35 data yang terklasifikasi dengan benar, terdapat kesalahan prediksi berjumlah 1 data yang masuk kedalam label *Positive* dan 30 data masuk kedalam label *Neutral*.

Pada hasil evaluasi metode *random forest* tanpa menggunakan *pos tagger* mendapatkan nilai *accuracy* 67.33 %, *precision* sebesar 74.88 %, *recall* sebesar 67.33 % dan *F-Score* sebesar 64.4 %. Hasil perhitungan *confusion matrix* pada metode *random forest* tanpa *pos tagger* dapat dilihat pada Tabel 4.10 dan Tabel 4.11.

Tabel 4.9 *Confusion Matrix* Metode *Random Forest* Tanpa *Pos Tagger*

Accuracy	67.33%
Precision	74.88%
Recall	67.33%
F-Score	64.40%

Tabel 4.10 Hasil Klasifikasi Metode *Random Forest* Tanpa *Pos Tagger*

	stem	label	pred_label_nb	pred_label_rf
0	childfree gittasav tunggu pensi	Positive	Positive	Neutral
1	duhh bingung pikir childfree gitasav tolol kay...	Negative	Negative	Neutral
2	minimal pikiran childfree	Neutral	Neutral	Neutral
3	allah jahat anak someday against orang milih c...	Negative	Negative	Negative
4	andai gitasav hidup paradis kampanye childfree	Neutral	Neutral	Neutral
...
246	salah gitasav ngomong childfree orang juang be...	Negative	Negative	Negative
247	ngebahas childfree kadang nyelutuk hah ngapain...	Negative	Negative	Positive
248	suruh childfree anak tega	Neutral	Neutral	Neutral
249	betul berat orgtua pantesan childfree liat sis...	Negative	Negative	Neutral
250	tuju childfree saiki goblok arek enom	Neutral	Neutral	Neutral

Tabel 4.11 Hasil Perbandingan Evaluasi Metode *Naïve Bayes Random Forest* Tanpa *Pos Tagger*

		Naive Bayes (%)	Random Forest (%)
0	Accuracy	69.72	67.33
1	Precision	71.86	74.88
2	Recall	69.72	67.33
3	F-Score	68.94	64.4

Pada Tabel 4.11 merupakan hasil perbandingan evaluasi metode *naïve bayes* dan *random forest* tanpa *pos tagger* yang mendapatkan hasil bahwa *accuracy*, *recall* dan *f1-score* metode *naïve bayes* lebih unggul dari metode *random forest*. Sedangkan, pada nilai *precision* metode *random forest* lebih unggul dari metode *naïve bayes*.

4.4 Percobaan Metode *Random Forest* Dan *Naïve Bayes* Dengan *Pos Tagger*

Dalam implementasi metode *naïve bayes* dan *random forest* dilakukan menggunakan bahasa pemrograman Python untuk proses klasifikasi pada dataset *childfree* pada Twitter. Dataset penelitian ini berupa pengambilan data dari *Twitter* yang berisi *tweet* opini masyarakat terkait *childfree*. Data yang digunakan dalam penelitian ini berjumlah 1309 *tweet* yang diambil dari tanggal 26 Februari 2023 hingga 08 Maret 2023. Dataset yang telah dikumpulkan dilakukan *load* dataset untuk melihat data tersebut yang hasilnya dapat dilihat pada Tabel 4.12.

Tabel 4.12 *Read Dataset*

	id	name	web	created_at	username	tweet	location
0	1.6E+18	lha.iki.lho	NaN	2023-03-	lha_iki_lhc	Childfree?	NaN
1	1.6E+18	Kore of C	NaN	2023-03-	saturnom	Happy W	NaN
2	1.6E+18	Habibi Ha	https://t.c	2023-03-	ibnurozii	@DitjenP	Depan Layar
3	1.6E+18	Dorr	NaN	2023-03-	ggakenna	@detikco	NaN
4	1.6E+18	-	NaN	2023-03-	vnylyp	@Askrlfes	NaN
...
1304	1.6E+18	Ajeng Kar	NaN	2023-02-	ajengkk	Aku emak	Malang, East Java
1305	1.6E+18	<i>girlfri</i>	https://t.c	2023-02-	ChacaVan	Kalian ter	Upper earth
1306	1.6E+18	rosaa	https://t.c	2023-02-	psiholic	@zhafirac	Universitas Gadjah Mada
1307	1.6E+18	Sapiens	NaN	2023-02-	Homo_sa	Twit ini ke	NaN
1308	1.6E+18	Yokanang	https://t.c	2023-02-	Yokanang	@Greesel	NaN

Dataset tersebut dilakukan *preprocessing* yang terdiri dari *case folding*, *tokenizing*, *filtering* dan *stemming* menggunakan *pos tagger* yang hasilnya dapat dilihat pada Gambar 4.13.

Tabel 4.13 *Preprocessing Pos Tagger*

	username	tweet	casefolding	tokenizing	filtering	stem	pos_tagger	extract_data	label
0	lha_iki_lho	Childfree?	childfree\nkalian	[childfree, kali	childfree kal	childfree k	[(childfree, FW)	childfree tidak bal	Negative
1	saternomeus	Happy W	happy womens	[happy, wom	happy wom	happy wor	((happy, FW), (happy womens d	Positive
2	ibnurozii	@DitjenP: huftt mungkin	[huftt, mungk	mungkin sel	mungkin s	[(mungkin, AD	mungkin sekaranc		Negative
3	ggakenna	@detikco se pandai apa	[sepandai, ap	sepandai ap	pandai ap	[(pandai, VB), (aroma pasti cium		Positive
4	vnlylp	@Askrfes zalim bgt ibunya	[zalim, begitu,	zalim ibunya	zalim ibu b	[(zalim, NN), (it	zalim ibu harus c		Positive
...
1290	ajengggkartik	Aku emak	aku emak emak	[aku, emak, er	aku emak er	aku emak e	((aku, PR), (em	emak emak anak	Neutral
1291	ChacaVarma	Kalian terl	kalian terlalu sit	[kalian, terlalu,	kalian terlalu	kalian terla	[(kalian, PR), (te	terlalu mulut gits	Negative
1292	psiholic	@zhafirac sendernya kep	[sendernya, k	sendernya k	sender kep	[(sender, NN), (sender kepo bjir c		Positive
1293	Homo_sapier	Twit ini ke	twit ini kejam k	[twit, ini, keja	twit kejam k	twit kejam	((twit, VB), (keja	kejam survival fitt	Negative
1294	Yokanang_	@Greesel kucingmu gak	[kucingmu, tic	kucingmu ti	kucing tida	[(kucing, NN), (kucing tidak child		Neutral

Dari Tabel 4.13 hasil *preprocessing* data *pos tagger* berjumlah 1295 data, dilakukan pengelompokkan jumlah label dari setiap label. Dalam penelitian ini menggunakan 3 jenis label yaitu Negative dengan total data 509, Neutral berjumlah 570 data dan Positive berjumlah 216 data.

Selanjutnya, dilakukan pembagian dataset menjadi 2 jenis data yaitu data *train* 80 % dan data *test* 20 % yang dapat dilihat pada Tabel 4.14 dan Tabel 4.15. Pada data *train* berjumlah 1036 data dan data *test* berjumlah 259 data.

Tabel 4.14 *Data Training Pada Pos Tagger*

	extract_data	label
0	hmm seperti childfree tidak	Negative
1	lamalama makin aneh netize	Negative
2	bukan gitav harus childfree	Positive
3	barakallahu laka whatsapp b	Neutral
4	hrusny ymir childfree tidak	Negative
...
1031	induk mungkin sudah pahar	Negative
1032	puja childfree bingung baca	Neutral
1033	dulu ingin jadi kaum selibat	Neutral
1034	mau childfree wkk	Neutral
1035	tidak siap punya anak tidak	Negative

Tabel 4.15 Data *Testing* Pada *Pos Tagger*

	extract_data	label
0	mereka sudah punah	Negative
1	cerdas orang childfree	Neutral
2	kamu lebih suka childf	Positive
3	dia paham childfree ac	Negative
4	benar kata gitasav me	Neutral
...
254	undang heri horeh bu	Neutral
255	turut serah orang mau	Negative
256	enak nyalahin tenaga	Neutral
257	childfree emang tidak	Negative
258	mau childfree tapi dia	Neutral

Pada masing-masing data dilakukan pembobotan kata menggunakan TF-IDF tradisional dengan memasukkan nilai delta. Dalam konteks ini, delta mengacu pada perubahan atau perbedaan dalam frekuensi relatif suatu kata antara dua dokumen atau lebih. yang hasilnya dapat dilihat pada Tabel 4.16 dan Tabel 4.17.

Tabel 4.16 Hasil *Delta TFIDF Train Data*

	aahhmama	abai	abangbang	abg	abisabisan	abissss	aborsi	about	absurd	abuse	...
D1	0	0	0	0	0	0	0	0	0	0	...
D2	0	0	0	0	0	0	0	0	0	0	...
D3	0	0	0	0	0	0	0	0	0	0	...
D4	0	0	0	0	0	0	0	0	0	0	...
D5	0	0	0	0	0	0	0	0	0	0	...
...
D1032	0	0	0	0	0	0	0	0	0	0	...
D1033	0	0	0	0	0	0	0	0	0	0	...
D1034	0	0	0	0	0	0	0	0	0	0	...
D1035	0	0	0	0	0	0	0	0	0	0	...
D1036	0	0	0	0	0	0	0	0	0	0	...

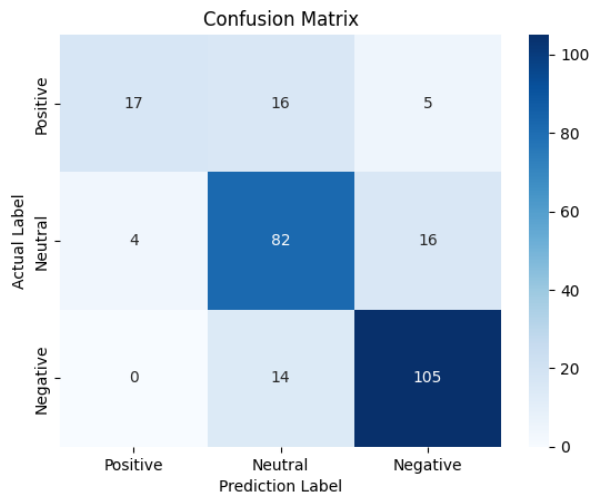
Tabel 4.17 Hasil *Delta TFIDF Testing Data*

	aahhmama	abai	abangbang	abg	abisabisan	abissss	aborsi	about	absurd	abuse	...
D1	0	0	0	0	0	0	0	0	0	0	...
D2	0	0	0	0	0	0	0	0	0	0	...
D3	0	0	0	0	0	0	0	0	0	0	...
D4	0	0	0	0	0	0	0	0	0	0	...
D5	0	0	0	0	0	0	0	0	0	0	...
...
D255	0	0	0	0	0	0	0	0	0	0	...
D256	0	0	0	0	0	0	0	0	0	0	...
D257	0	0	0	0	0	0	0	0	0	0	...
D258	0	0	0	0	0	0	0	0	0	0	...
D259	0	0	0	0	0	0	0	0	0	0	...

Selanjutnya, dilakukan pembentukan model *naïve bayes* dan *random forest* (*Pos Tagger*) untuk melatih model menggunakan data *train* dan data *test* serta dilakukan evaluasi menggunakan *confusion matrix* untuk mengetahui performa model dalam klasifikasi. Berikut merupakan hasil *confusion matrix* metode *naïve bayes* dan *random forest* menggunakan *pos tagger*.

1. Hasil Evaluasi Metode *Naïve Bayes*

Metode *naïve bayes* dilakukan evaluasi menggunakan *confusion matrix* pada data *test* yang mendapatkan hasil pada Gambar 4.39.



Gambar 4.39 *Confusion Matrix* Metode *Naïve Bayes* Menggunakan *Pos Tagger*

Pada Gambar 4.39 dapat dijelaskan bahwa metode *naïve bayes* menggunakan *pos tagger* diperoleh hasil prediksi label *Positive* berjumlah 38 data yang diuji, terdapat 17 data yang terklasifikasi dengan benar dan terdapat kesalahan prediksi

berjumlah 16 data yang masuk kedalam label *Neutral* dan 5 data masuk kedalam label *Negative*. Sedangkan, pada label *Neutral* dengan jumlah data 102 yang diuji, terdapat 82 data yang terklasifikasi dengan benar, kesalahan prediksi berjumlah 4 data yang masuk kedalam label *Positive* dan 16 data pada label *Negative*. Pada label *Negative* dengan jumlah data 119 data yang diuji, terdapat 105 data yang terklasifikasi dengan benar, terdapat kesalahan prediksi berjumlah 14 data masuk kedalam label *Neutral*.

Tabel 4.18 Hasil *Confusion Matrix* Metode *Naïve Bayes* Menggunakan *Pos Tagger*

Accuracy	78.76%
Precision	79.00%
Recall	78.76%
F-Score	78.02%

Pada Tabel 4.18 hasil evaluasi metode *naïve bayes* tanpa menggunakan *pos tagger* mendapatkan nilai *accuracy* 78.76 %, *precision* sebesar 79.0 %, *recall* sebesar 78.76 % dan *F-Score* sebesar 78.02 %.

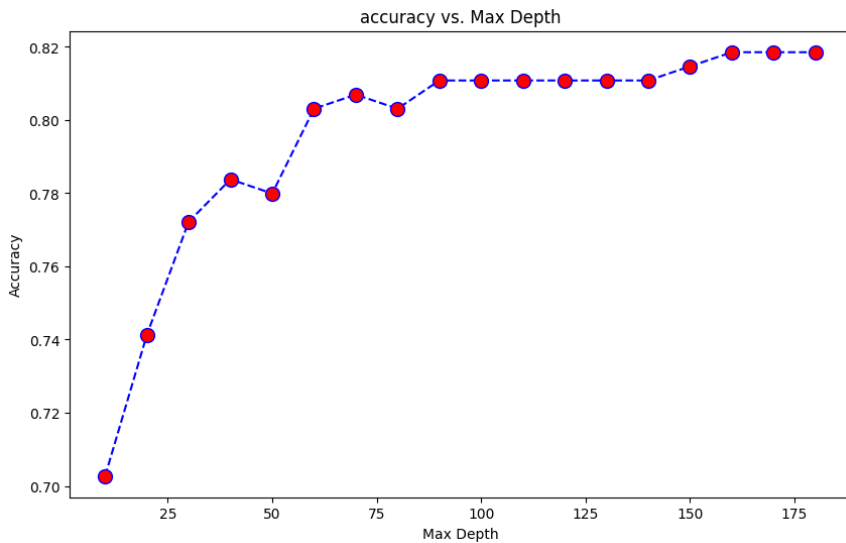
Tabel 4.19 Hasil Klasifikasi *Naïve Bayes* Menggunakan *Pos Tagger*

	extract_data	label	pred_label_nb
0	mereka sudah punah childfree	Negative	Neutral
1	cerdas orang childfree buat u	Neutral	Positive
2	kamu lebih suka childfree swe	Positive	Positive
3	dia paham childfree ada janta	Negative	Negative
4	benar kata gitav mending c	Neutral	Neutral
...
254	undang heri horeh bunga suc	Neutral	Neutral
255	turut serah orang mau childfr	Negative	Negative
256	enak nyalahin tenaga didik en	Neutral	Neutral
257	childfree emang tidak buruk k	Negative	Negative
258	mau childfree tapi dia breedki	Neutral	Neutral

Pada Tabel 4.19 merupakan hasil klasifikasi *naïve bayes* tanpa *post tagger* dengan total data 259 data beserta labelnya.

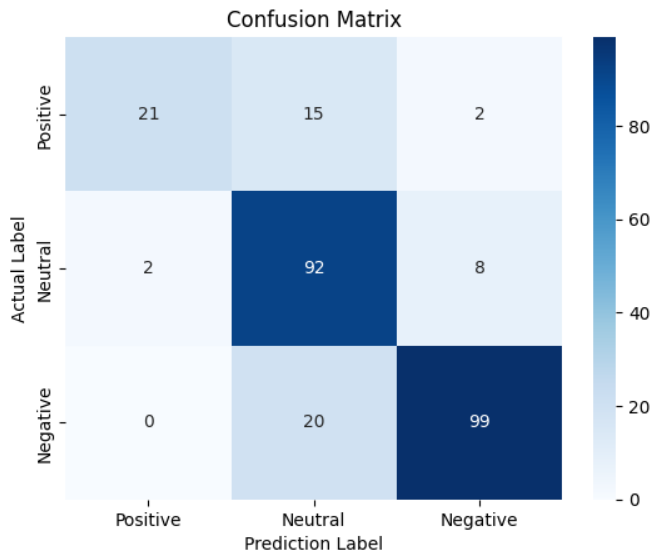
2. Hasil Evaluasi Metode *Random Forest*

Dalam metode *random forest* dilakukan visualisasi plot untuk mendapatkan hasil *maximum accuracy* dan nilai *max dept*. *Maximum accuracy* sebesar 0.8185 at *max depth* = 160 dan nilai *max depth* terbaik adalah 160. Hasil visualisasi plot dapat dilihat pada Gambar 4.40.



Gambar 4.40 Hasil Visualisasi Plot Metode *Random Forest* Menggunakan *Pos Tagger*

Kemudian dilakukan evaluasi menggunakan *confusion matrix* yang didapatkan hasil pada Gambar 4.41.



Gambar 4.41 *Confusion Matrix* Metode *Random Forest* Menggunakan *Pos Tagger*

Pada Gambar 4.41 dapat dijelaskan bahwa metode *random forest* menggunakan *pos tagger* diperoleh hasil prediksi label *Positive* berjumlah 38 data yang diuji, terdapat 21 data yang terklasifikasi dengan benar dan terdapat kesalahan prediksi berjumlah 15 data yang masuk kedalam label *Neutral* dan 2 data masuk kedalam label *Negative*. Sedangkan, pada label *Neutral* dengan jumlah data 102 yang diuji, terdapat 92 data yang terklasifikasi dengan benar, kesalahan prediksi berjumlah 2 data pada label *Positive* dan 8 data pada label *Negative*. Pada label *Negative* dengan jumlah data 119 data yang diuji, terdapat 99 data yang terklasifikasi dengan benar, terdapat kesalahan prediksi berjumlah 20 data masuk kedalam label *Neutral*.

Pada hasil evaluasi metode *random forest* menggunakan menggunakan *pos tagger* mendapatkan nilai *accuracy* 81.85 %, *precision* sebesar 83.66 %, *recall* sebesar 81.85 % dan *F-Score* sebesar 81.65 %. Hasil perhitungan *confusion matrix* pada metode *random forest* menggunakan *pos tagger* dapat dilihat pada Tabel 4.21 dan Tabel 4.22.

Tabel 4.20 Hasil *Confusion Matrix* Metode *Random Forest* Menggunakan *Pos Tagger*

Accuracy	81.85%
Precision	83.66%
Recall	81.85%
F-Score	81.65%

Tabel 4.21 Hasil Klasifikasi Metode *Random Forest* Menggunakan *Pos Tagger*

	extract_data	label	pred_label_nb	pred_label_rf
0	mereka sudah punah	Negative	Neutral	Neutral
1	cerdas orang childfree	Neutral	Negative	Neutral
2	kamu lebih suka child	Positive	Positive	Positive
3	dia paham childfree a	Negative	Negative	Negative
4	benar kata gitasav me	Neutral	Negative	Neutral
...
254	undang heri horeh bu	Neutral	Neutral	Neutral
255	turut serah orang ma	Negative	Negative	Negative
256	enak nyalahin tenaga	Neutral	Neutral	Neutral
257	childfree emang tidak	Negative	Negative	Negative
258	mau childfree tapi dia	Neutral	Neutral	Neutral

Tabel 4.22 Hasil Perbandingan Evaluasi Metode *Naïve Bayes* *Random Forest* Menggunakan *Pos Tagger*

		Naive Bayes (%)	Random Forest (%)
0	Accuracy	78.76	81.85
1	Precision	79	83.66
2	Recall	78.76	81.85
3	F1-Score	78.02	81.65

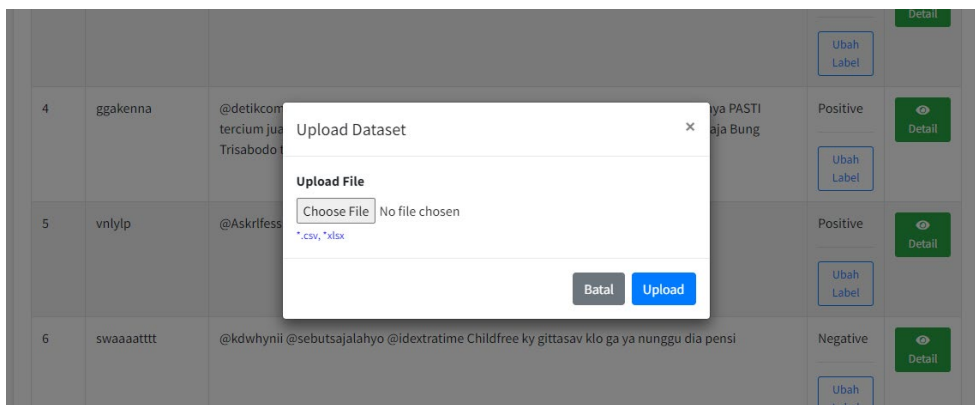
Pada Tabel 4.22 merupakan hasil perbandingan evaluasi metode *naïve bayes* dan *random forest* menggunakan *pos tagger* yang mendapatkan hasil bahwa metode *random forest* lebih unggul dari metode *naïve bayes*.

4.5 Hasil Implementasi Aplikasi

Pada tahap ini dilakukan implementasi algoritma *Naive Bayes* dan *Random Forest* ke dalam bentuk website untuk menampilkan Hasil analisis sentimen masyarakat terhadap opini di *Twitter*.

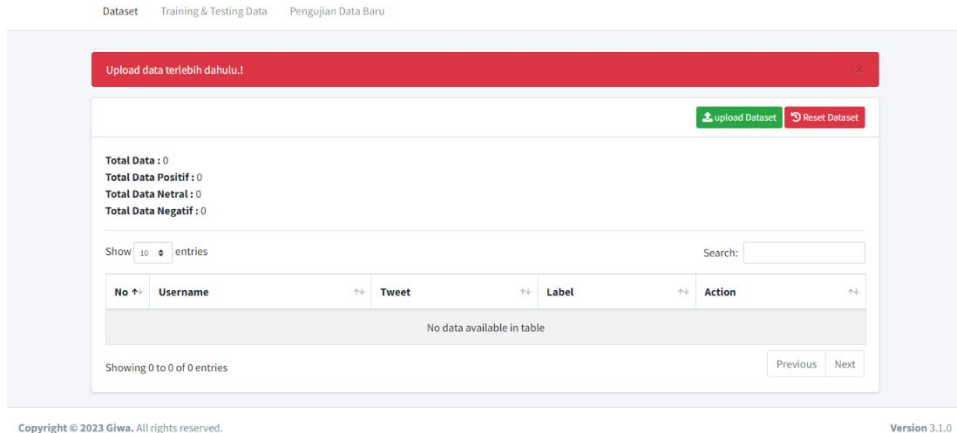
4.5.1 Implementasi Halaman Analisis Data Sentimen

Implementasi aplikasi tampilan analisis data sentimen, lakukan import data dari hasil scrapping untuk mengetahui hasil analisis sentimeen dengan metode *naive bayes* dan *random forest*. Berikut Tampilan Import file, dapat dilihat pada Gambar 4.42.



Gambar 4.42 Tampilan *Upload File Dataset*

Jika pengguna belum mengunggah dataset, muncul pesan pop-up yang menyatakan "Mohon unggah dataset terlebih dahulu," tampilan ini dapat dilihat pada Gambar 4.43. Pesan ini dirancang untuk memberikan pengingat kepada pengguna agar melengkapi langkah-langkah yang diperlukan sebelum melanjutkan, memastikan bahwa proses berjalan sesuai dengan prosedur yang ditetapkan. Dengan adanya pesan tersebut, pengguna dapat dengan cepat mengidentifikasi kebutuhan untuk mengunggah dataset sebelum melanjutkan ke langkah atau fungsi berikutnya dalam aplikasi atau sistem yang digunakan.



Gambar 4.43 Tampilan *Pop-up*

4.5.2 Implementasi Halaman Dataset

Halaman ini menampilkan dataset yang telah diunggah, yang mencakup kolom-kolom No, username, tweet, dan label. Total data yang terdapat dalam dataset ini mencapai 1295 entri. Adapun rinciannya, terdapat 216 entri data positif, 570 entri data netral, dan 509 entri data negatif. Informasi ini memberikan gambaran menyeluruh tentang komposisi dataset yang digunakan dalam analisis atau aplikasi tertentu. Tampilan dataset dapat ditemukan pada Gambar 4.44, memberikan visualisasi yang jelas terkait struktur dan distribusi data yang menjadi dasar analisis atau pemrosesan selanjutnya.

Data berhasil di simpan!

upload Dataset
Reset Dataset

Total Data : 1295
Total Data Positif : 216
Total Data Netral : 570
Total Data Negatif : 509

Show entries Search:

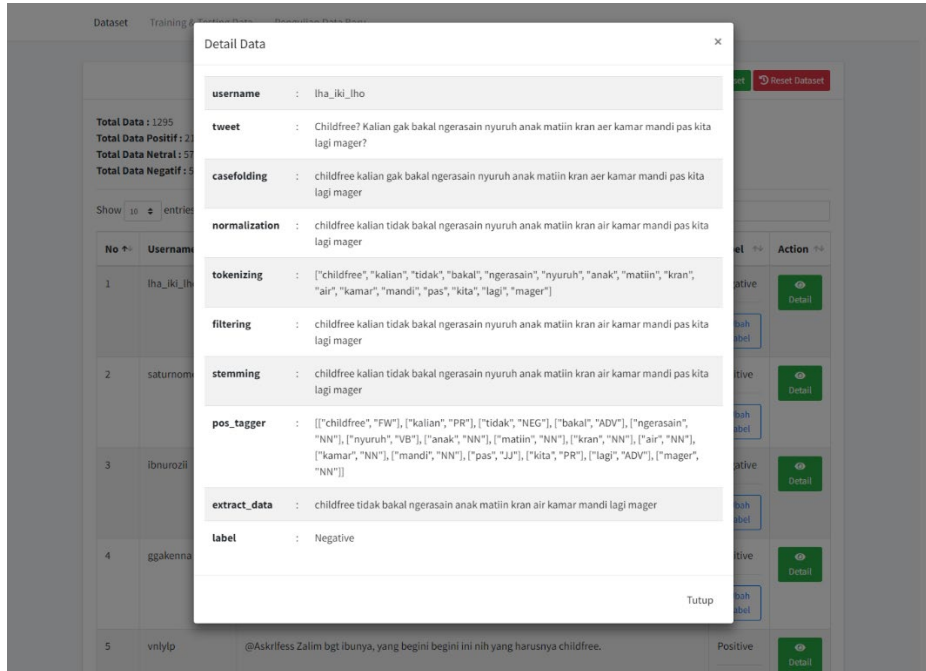
No ↕	Username ↕	Tweet ↕	Label ↕	Action ↕
1	lha_ki_lho	Childfree? Kalian gak bakal ngerasain nyuruh anak matiin kran aer kamar mandi pas kita lagi mager?	Negative	Detail Ubah Label
2	saturnomeus	Happy Women's Day untuk single mothers, perempuan yang fokus karir, yang ingin childfree, transpuan, perempuan lgbtq+ dan semua perempuan yang berjuang untuk kebebasan! ❤️❤️❤️	Positive	Detail Ubah Label
3	ibnurozil	@DitjenPajakRI Huftt mungkin skrng dipikrannya childfree tidak terlalu buruk juga ?	Negative	Detail Ubah Label
4	ggakenna	@detikcom Sepandai apapun kalian menyimpan bangkai, cepat-lambat aroma busuknya PASTI tercium jua. Iya bisa lewat mana saja. Makanya kalo niat mau jadi penjahat, childfree saja Bung Trisabodo teuing?	Positive	Detail Ubah Label
5	vnlylp	@Askrlfess Zalim bgt ibunya, yang begini begini ini nih yang harusnya childfree.	Positive	Detail Ubah Label
6	swaaaatttt	@kdwhyinii @sebutsajalahyo @idextratime Childfree ky gittasav klo ga ya nunggu dia pensi	Negative	Detail Ubah Label
7	llon__	@yellowwillowow @tanyakanri gak ngantuk childfree, tapi lebih sedikasihnya aja hehe. mikir klo mau punya anak takut akunya gak siap secara mental dan finansial, tapi pengen punya anak. jadi ya yaudah sedikasihnya aja	Negative	Detail Ubah Label
8	Badrebellions	Sepertinya childfree memang tidak seburuk itu. Untuk kategori negara yang populasinya banyak tapi gak berkontribusi banget childfree adalah pilihan yang menarik.	Neutral	Detail Ubah Label
9	heyraaraa	dear orang jahat, kenapa bapak dan ibumu tidak childfree aja??	Negative	Detail Ubah Label
10	hiirael	@jokoanwar " Andai waktu itu aku childfree "	Neutral	Detail Ubah Label

Showing 1 to 10 of 1,295 entries
Previous
1
2
3
4
5
...
130
Next

Gambar 4.44 Tampilan Dataset

4.5.3 Implementasi Halaman Detail Data

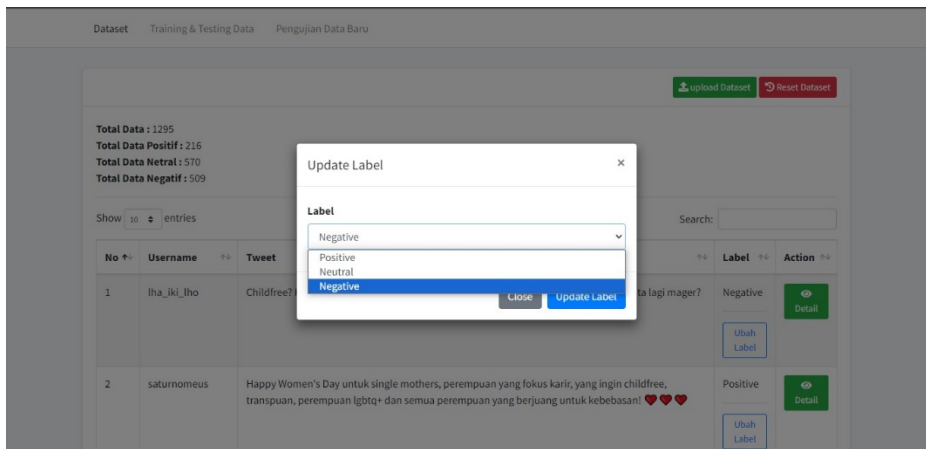
Halaman ini akan menampilkan seluruh atribut dan informasi yang terkait dengan entri tersebut, seperti teks *tweet*, label sentimen, dan atribut lainnya, dapat dilihat pada Gambar 4.45.



Gambar 4.45 Tampilan Detail Data

4.5.4 Implementasi Update Label

Dalam melakukan update label pada sistem pengguna dapat melakukan secara manual, dimana pengguna dapat memilih label positif, negatif, atau netral. Berikut tampilan update label dapat dilihat pada Gambar 4.46.



Gambar 4.46 Tampilan *Update Label*

4.5.5 Implementasi Halaman Hasil Klasifikasi Menggunakan Data Training dan Testing

Halaman Hasil Klasifikasi Menggunakan Data Training dan Testing merupakan bagian penting dalam proses analisis data dan evaluasi model. Halaman ini menyajikan hasil klasifikasi yang diperoleh dari model yang telah dilatih menggunakan data training dan diuji dengan data testing, dapat dilihat pada Gambar 4.47.

Hasil Analisis Sentimen Menggunakan Naive Bayes dan Random Forest

DATA TRAIN

Total Data Negatif : 390
Total Data Netral : 468
Total Data Positif : 178

Show entries Search:

No ↕	Extract Data ↕	Label ↕
1	hmm childfree tidak	Negative
2	makin netizen sebel childfree anak-anak terus comment gita	Negative
3	bukan gitasav harus childfree	Positive
4	barakallahu laka whatsapp baraka alaika wajamaa finance sahabat allah selalu istri bakti allah orang amin tidak childfree sihhhh	Neutral
5	hrusny ymir childfree tidak	Negative
6	kali liat orang childfree sikap anak sekitar toxic parents	Negative
7	sudah gin jangan nyalahin orang mau childfree ges	Neutral
8	childfree orang bukan ideologi bisa anak tidak masalah	Negative
9	gita mau childfree kontroversi bisa teknik nuklir	Neutral
10	bapak dulu tidak childfree	Negative

Showing 1 to 10 of 1,036 entries
[Previous](#)
[1](#)
[2](#)
[3](#)
[4](#)
[5](#)
[...](#)
[104](#)
[Next](#)

DATA TEST

Total Data Negatif : 119
Total Data Netral : 102
Total Data Positif : 38

Show entries Search:

No ↕	Extract Data ↕	Label ↕
1	sudah punah childfree	Negative
2	orang childfree tingkat kualitas amp kebahagiaan	Neutral
3	lebih childfree sweet child omine	Positive
4	paham childfree jantan nyamperin tidak hamil lagi	Negative
5	kata gitasav childfree nyusahin orang	Neutral
6	kucing hrusnya childfree	Positive
7	biarin guys childfree childfree emang anak tidak kata dunia sudah jawab ortu	Negative
8	childfree lip	Neutral
9	tidak childfree	Negative
10	faktor childfree kian hari kian bukan ngikutin gengsi	Neutral

Showing 1 to 10 of 259 entries
[Previous](#)
[1](#)
[2](#)
[3](#)
[4](#)
[5](#)
[...](#)
[26](#)
[Next](#)

Gambar 4.47 Tampilan Hasil Data *Training* dan *Testing*

Sedangkan hasil klasifikasi dari kedua buah metode dapat dilihat pada Gambar 4.48.

Hasil Analisis Sentimen Menggunakan Naive Bayes dan Random Forest

DATA TRAIN

Total Data Negatif : 390
 Total Data Netral : 468
 Total Data Positif : 178

Show 10 entries

Search:

No ↑↓	Extract Data	Label ↑↓
1	hmm childfree tidak	Negative
2	makin netizen sebel childfree anak-anak terus comment gita	Negative
3	bukan gitasav harus childfree	Positive
4	barakallahu laka whatsapp baraka alaika wajamaa finance sahabat allah selalu istri bakti allah orang amin tidak childfree sihhhh	Neutral
5	hrusny ymir childfree tidak	Negative
6	kali liat orang childfree sikap anak sekitar toxic parents	Negative
7	sudah gin jangan nyalahin orang mau childfree ges	Neutral
8	childfree orang bukan ideologi bisa anak tidak masalah	Negative
9	gita mau childfree kontroversi bisa teknik nuklir	Neutral
10	bapak dulu tidak childfree	Negative

Showing 1 to 10 of 1,036 entries

Previous 1 2 3 4 5 ... 104 Next

DATA TEST

Total Data Negatif : 119
 Total Data Netral : 102
 Total Data Positif : 38

Show 10 entries

Search:

No ↑↓	Extract Data	Label ↑↓
1	sudah punah childfree	Negative
2	orang childfree tingkat kualitas amp kebahagiaan	Neutral
3	lebih childfree sweet child omine	Positive
4	paham childfree jantan nyamperin tidak hamil lagi	Negative
5	kata gitasav childfree nyusahin orang	Neutral
6	kucing hrusnya childfree	Positive
7	biarin guys childfree childfree emang anak tidak kata dunia sudah jawab ortu	Negative
8	childfree lip	Neutral
9	tidak childfree	Negative
10	faktor childfree kian hari kian bukan ngikutin gengsi	Neutral

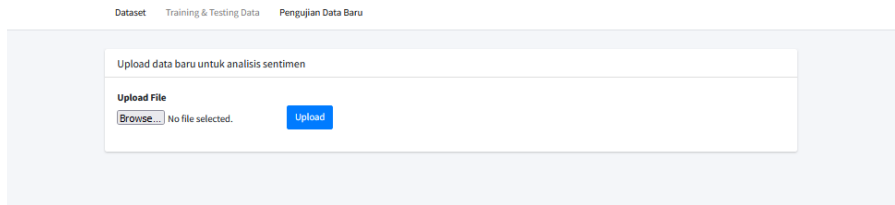
Showing 1 to 10 of 259 entries

Previous 1 2 3 4 5 ... 26 Next

Gambar 4.48 Tampilan Hasil Klasifikasi

4.5.6 Implementasi Halaman Hasil Klasifikasi Menggunakan Data Baru

Halaman Hasil Klasifikasi Menggunakan Dataset Baru adalah tampilan yang bertujuan untuk menampilkan hasil dari klasifikasi data yang telah dianalisis menggunakan model yang telah dilatih sebelumnya, namun dengan dataset yang baru, dapat dilihat pada Gambar 4.49.



Gambar 4.49 Tampilan Halaman *Dataset* Baru

4.6 Pengujian Hasil Pelabelan

Pengujian hasil pelabelan melibatkan pengujian pada analisis sentimen dengan dua teknik pelabelan dataset yang berbeda, yakni menggunakan corpus dan pelabelan manual oleh manusia. Dalam pengujian ini data yang digunakan sebanyak 260 data. Berikut hasil akurasi menggunakan *corpus* dan *manual* pada analisis masyarakat terhadap *childfree* dengan menggunakan metode *random forest* dan *naïve bayes*.

a) Hasil Pelabelan Menggunakan *Corpus*

Hasil pelabelan menggunakan *corpus* merupakan teknik yang digunakan untuk mencocokkan data dalam menentukan kelas labelnya dengan kumpulan data dalam bentuk teks. Adapun hasil akurasi yang diperoleh pada pelabelan *corpus* dapat dilihat pada gambar 4.35 untuk metode *naïve bayes* dan Gambar 4.50 untuk metode *random forest*.

Naive Bayes

Total Data Prediksi Negatif : 26
Total Data Prediksi Netral : 25
Total Data Prediksi Positif : 2

Show entries Search:

No ↑↓	Extract Data ↑↓	Label Aktual ↑↓	Label Prediksi ↑↓
1	alesan ingin childfree since lalu belum ibu belum orang tidak bisa ngebesarin anak nyakitin anak tidak bisa handle	Neutral	Negative
2	yaiyalah ego tidak relate orang emang tidak ingin anak nama childfree bagaimana sudah	Negative	Negative
3	anak childfree	Neutral	Neutral
4	belum calon belum sudah duluan childfree dunia luar sekarang lagi tidak mau desa kota sudah tidak tempat	Negative	Negative
5	telah program anak cukup program childfree	Neutral	Negative
6	simpatisan childfree kampanye	Positive	Neutral
7	informasi masjid jamaah childfree pen nasi kotak susul bocil terus	Neutral	Negative
8	mau gapapa sewa gbk mau terus cateringnya mau bakso afung terus honeymoon bulan soal tidak bisa terus childfree mau anak	Positive	Negative
9	emang childfree bisa	Positive	Positive
10	harus childfree aja anaknya vitri	Positive	Neutral

Showing 1 to 10 of 53 entries

Accuracy = 71.7 %
Precision = 76.72 %
Recall = 71.7 %
F-Score = 70.27 %

Gambar 4.50 Hasil Akurasi *Naive Bayes*

Random Forest			
Total Data Prediksi Negatif : 21			
Total Data Prediksi Netral : 28			
Total Data Prediksi Positif : 4			
Show 10 entries		Search: <input type="text"/>	
No ↑	Extract Data	Label Aktual	Label Prediksi
1	alesan ingin childfree since lalu belum ibu belum orang tidak bisa ngebesarin anak nyakitin anak tidak bisa handle	Neutral	Negative
2	yaiyalah ego tidak relate orang emang tidak ingin anak nama childfree bagaimana sudah	Negative	Negative
3	anak childfree	Neutral	Neutral
4	belum caton belum sudah duluan childfree dunia luar sekarang lagi tidak mau desa kota sudah tidak tempat	Negative	Negative
5	telah program anak cukup program childfree	Neutral	Neutral
6	simpatisan childfree kampanye	Positive	Neutral
7	informasi masjid jamaah childfree pen nasi kotak susul bocil terus	Neutral	Neutral
8	mau gapapa sewa gbk mau terus cateringnya mau bakso afung terus honeymoon bulan soal tidak bisa terus childfree mau anak	Positive	Negative
9	emang childfree bisa	Positive	Positive
10	harus childfree aja anaknya vitri	Positive	Positive
Showing 1 to 10 of 53 entries		Previous 1 2 3 4 5 6 Next	
Accuracy = 81.13 % Precision = 83.02 % Recall = 81.13 % F-Score = 81.01 %			

Gambar 4.51 Hasil Akurasi *Random Forest*

Berdasarkan Gambar 4.51 dapat disimpulkan bahwa metode *naïve bayes* memperoleh hasil akurasi sebesar 71.7% dengan nilai Precision = 76.32%, Recall = 71.7%, dan F-Score = 70.89%. Sementara itu, pada Gambar 4.38 dapat disimpulkan bahwa metode *random forest* menghasilkan akurasi sebesar 81.13%, Precision = 82.73%, Recall = 81.137%, dan F-Score = 80.54%.

b) Hasil Pelabelan Menggunakan Manual

Hasil pelabelan menggunakan manual merupakan teknik yang digunakan secara manual dimana pelabelan ditentukan berdasarkan nalar manusia. Adapun hasil akurasi yang diperoleh pada pelabelan *corpus* dapat dilihat pada gambar 4.52 untuk metode *naive bayes* dan Gambar 4.53 untuk metode *random forest*.

Naive Bayes

Total Data Prediksi Negatif : 25
Total Data Prediksi Netral : 26
Total Data Prediksi Positif : 2

Show entries Search:

No ↑	Extract Data ↑	Label Aktual ↑	Label Prediksi ↑
1	alesan ingin childfree since lalu belum ibu belum orang tidak bisa ngebesarin anak nyakitin anak tidak bisa handle	Neutral	Negative
2	yaiyalah ego tidak relate orang emang tidak ingin anak nama childfree bagaimana sudah	Negative	Negative
3	anak childfree	Neutral	Neutral
4	belum calon belum sudah duluan childfree dunia luar sekarang lagi tidak mau desa kota sudah tidak tempat	Negative	Negative
5	telah program anak cukup program childfree	Neutral	Negative
6	simpatisan childfree kampanye	Positive	Neutral
7	informasi masjid jamaah childfree pen nasi kotak susul bocil terus	Neutral	Negative
8	mau gapapa sewa gbk mau terus cateringnya mau bakso afung terus honeymoon bulan soal tidak bisa terus childfree mau anak	Positive	Neutral
9	emang childfree bisa	Positive	Neutral
10	harus childfree aja anaknya vitri	Positive	Neutral

Showing 1 to 10 of 53 entries

Accuracy = 69.81 %
Precision = 75.3 %
Recall = 69.81 %
F-Score = 67.92 %

Gambar 4.52 Hasil Akurasi *Naive Bayes*

Random Forest

Total Data Prediksi Negatif : 21
 Total Data Prediksi Netral : 28
 Total Data Prediksi Positif : 4

Show entries Search:

No ↕	Extract Data	Label Aktual ↕	Label Prediksi ↕
1	alesan ingin childfree since lalu belum ibu betum orang tidak bisa ngebesarin anak nyakitin anak tidak bisa handle	Neutral	Negative
2	yaiyalah ego tidak relate orang emang tidak ingin anak nama childfree bagaimana sudah	Negative	Negative
3	anak childfree	Neutral	Neutral
4	belum calon belum sudah duluan childfree dunia luar sekarang lagi tidak mau desa kota sudah tidak tempat	Negative	Negative
5	telah program anak cukup program childfree	Neutral	Neutral
6	simpatisan childfree kampanye	Positive	Neutral
7	informasi masjid jamaah childfree pen nasi kotak susul bocil terus	Neutral	Neutral
8	mau gapapa sewa gbk mau terus cateringnya mau bakso afung terus honeymoon bulan soal tidak bisa terus childfree mau anak	Positive	Negative
9	emang childfree bisa	Positive	Positive
10	harus childfree aja anaknya vitri	Positive	Positive

Showing 1 to 10 of 53 entries Previous **1** 2 3 4 5 6 Next

Accuracy = 79.25 %
 Precision = 82.12 %
 Recall = 79.25 %
 F-Score = 78.99 %

Gambar 4.53 Hasil Akurasi *Random Forest*

Berdasarkan Gambar 4.33 dapat disimpulkan bahwa metode *naïve bayes* memperoleh hasil akurasi sebesar 69.81 % dengan Precision = 75.3 %, Recall = 69.81 %, F-Score = 67.92 %. Sedangkan pada Gambar 4.40 dapat disimpulkan bahwa hasil akurasi menggunakan *random forest* memperoleh nilai akurasi sebesar 79.25 %, Precision = 82.12 %, Recall = 79.25 %, F-Score = 78.99 %. Hasil ini disendirikan dengan tampilan di web

4.7 Pembahasan

Dalam penelitian ini, analisis sentimen masyarakat terhadap opini di *Twitter* telah dilakukan. Data yang digunakan terdiri dari 1309 *tweet* yang dikumpulkan dari tanggal 26 Februari 2023 hingga 08 Maret 2023. Data tersebut dibagi menjadi dua bagian: data *training*, yang digunakan untuk melatih model analisis sentimen, dan data *testing*, yang digunakan untuk menguji performa model. Terdapat enam sampel data yang mewakili berbagai kelas, dan dataset ini dibagi menjadi dua bagian, yaitu train (80%) dan test (20%). Hasil evaluasi menggunakan Pos Tagger antara metode *Naïve Bayes* dan *Random Forest* terhadap opini masyarakat di *Twitter* menunjukkan bahwa *Random Forest* memiliki akurasi yang lebih tinggi, yaitu 81.85%, dibandingkan

dengan *Naïve Bayes* yang memiliki akurasi sebesar 78.75%. Artinya, *Random Forest* berhasil mencapai akurasi yang lebih tinggi dalam mengklasifikasikan sentimen masyarakat terhadap opini di *Twitter* dibandingkan dengan metode *Naïve Bayes*. Sementara hasil evaluasi tanpa Pos Tagger menggunakan *naïve bayes* dan *random forest*, menghasilkan nilai akurasi sebesar 69.72% untuk *Naïve Bayes*, dan nilai akurasi *Random Forest* sebesar 67.33%. Artinya, *Naïve Bayes* berhasil mencapai akurasi yang lebih tinggi dalam mengklasifikasikan sentimen masyarakat terhadap opini di *Twitter* dibandingkan dengan metode *Random Forest*. Berikut merupakan hasil *confusion matrix* dari proses menggunakan pos tagger dan tanpa pos tagger, dapat dilihat pada Tabel 4.23.

Tabel 4.23 Hasil *Confusion Matrix* Metode *Naive Bayes* Dan *Random Forest*

	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>f-score</i>
<i>Naïve Bayes</i>				
Tanpa Pos Tagger	69.72%	71.86%	69.72%	68.94%
Menggunakan Pos Tagger	78.76%	79.00%	78.76%	78.02%
<i>Random Forest</i>				
Tanpa Pos Tagger	67.33%	74.88%	67.33%	64.40%
Menggunakan Pos Tagger	81.85%	83.66%	81.85%	81.65%

Selain itu pada penelitian ini peneliti melakukan pengujian terhadap dua teknik pelabelan menggunakan *corpus* dan pelabelan manual oleh manusia. Dalam pengujian ini data yang digunakan sebanyak 260 data. Berdasarkan pengujian tersebut diperoleh hasil pelabelan menggunakan *corpus* dengan metode *naïve bayes* memperoleh akurasi sebesar 71.7% dan metode *random forest* memperoleh akurasi sebesar 81.13%, sedangkan dalam pelabelan manual metode *naïve* memperoleh akurasi sebesar 69.81 %, sedangkan metode *random forest* memperoleh akurasi sebesar 79.25 %. Dapat dilihat perbandingannya pada Tabel 4.24 berikut.

Tabel 4.24 Perbandingan Pelabelan Manual dan Menggunakan *Corpus*

Pelabelan <i>Corpus</i>		Pelabelan Manual	
<i>Random Forest</i>	<i>Naïve Bayes</i>	<i>Random Forest</i>	<i>Naïve Bayes</i>
79.25%	69.81%	81.13%	71.70%

Berdasarkan uraian tersebut, dapat disimpulkan bahwa penelitian ini fokus pada analisis sentimen masyarakat terhadap opini di *Twitter* menggunakan dua metode pelabelan dataset, yaitu menggunakan corpus dan pelabelan manual oleh manusia. Hasil pengujian menunjukkan bahwa *Random Forest* memiliki akurasi lebih baik dari metode *Naïve Bayes* dalam pelabelan menggunakan *corpus* maupun pelabelan manual. Sedangkan dari hasil evaluasi menggunakan *Pos Tagger* menunjukkan bahwa *Random Forest* memiliki akurasi lebih tinggi dalam mengklasifikasikan sentimen masyarakat terhadap opini di *Twitter* dibandingkan dengan metode *Naïve Bayes*. Tanpa menggunakan *Pos Tagger*, *Naïve Bayes* kembali berhasil mencapai akurasi yang lebih tinggi daripada *Random Forest* dalam mengklasifikasikan sentimen tersebut.