BAB 3
METODE PENELITIAN

Sebagai langkah awal, penelitian ini mengumpulkan data tentang Presiden dan Wakil Presiden terpilih yang tersedia di aplikasi X., selanjutnya melakukan pengolahan data yang telah dikumpulkan untuk menentukan sentimen yang akurat, sehingga informasi yang diperoleh menjadi lebih bernilai. Berikut paparan mengenai bahan, peralatan, metodologi penelitian, dan tahapannya.

3.1 BAHAN DAN ALAT PENELITIAN

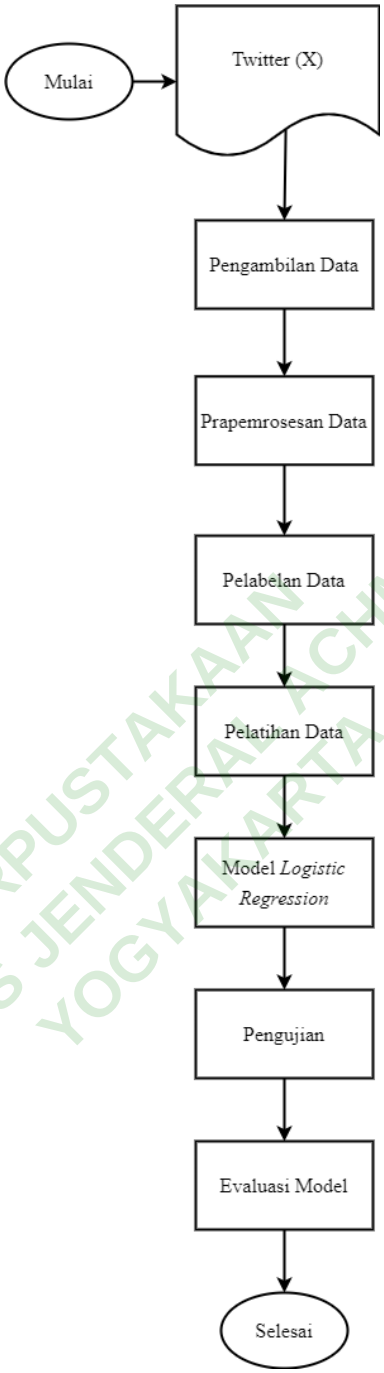
Data diambil dari aplikasi X yang memuat kata kunci "prabowo" dan "gibran_*tweet*". Data ini relevan dengan fokus penelitian yang meneliti persepsi publik setelah penetapan Presiden dan Wakil Presiden terpilih.

Penelitian ini dilaksanakan dengan menggunakan laptop yang memiliki spesifikasi memadai untuk menunjang kelancaran proses pengolahan data dan koneksi internet. Hal ini memastikan kelancaran dan efisiensi dalam pelaksanaan penelitian. Dalam penelitian ini, beberapa perangkat lunak dan sistem operasi (*OS*) digunakan, antara lain:

1. *OS* Windows 10 versi *64-bit*.
2. *Programming Language Python* versi 3.12.1.
3. *Microsoft Office Excel* 2019.
4. *Google Colab*.

3.2 JALAN PENELITIAN

Riset ini memanfaatkan aplikasi *Google Colab* dan pustaka *Tweet Harvest* untuk mengumpulkan data. Data yang terkumpul kemudian disimpan dalam format file *CSV* dan dimodelkan menggunakan pustaka bahasa pemrograman *Python*.



Gambar 3.1 Alur Penelitian

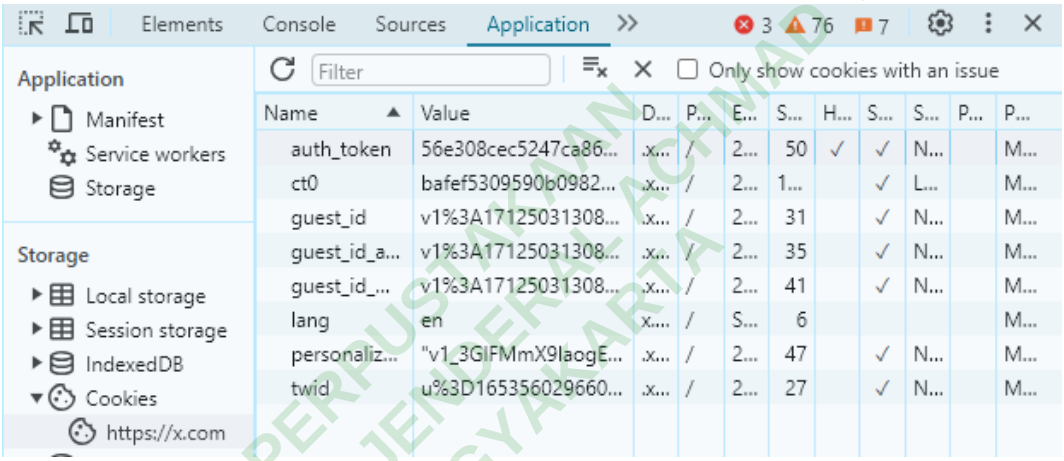
3.2.1 Pengambilan Data

Pengambilan data dari aplikasi X untuk mendapatkan data *tweet* tentang Presiden dan Wakil Presiden terpilih dengan menggunakan pustaka *Tweet Harvest* dan dieksekusi di *Google Colab*. Data *tweet* yang diambil yaitu *tweet* yang

berkaitan dengan kata kunci “prabowo” dan “gibran_tweet” selama periode 24 April – 30 April 2024 dengan jumlah 39.637 data. Pengambilan data memerlukan 2 tahapan yaitu:

3.2.1.1 Mendapatkan Token Autentikasi Aplikasi X

Token autentikasi aplikasi X adalah string unik yang memungkinkan aplikasi pihak ketiga untuk mengakses akun aplikasi X dengan cara yang aman dan terkendali. *Token* ini menyediakan aplikasi dengan akses terbatas ke data dan fungsi akun aplikasi X, seperti membaca *tweet*.



Gambar 3.2 Tampilan Token Autentikasi Aplikasi X

Pasca autentikasi akun aplikasi X, *Token* didapatkan melalui *inspect* pada browser *Chrome* pada bagian *Cookies*.

3.2.1.2 Install Node JS

Karena pustaka *Tweet Harvest* dibuat dengan *Node JS*, maka perlu *install Node JS* terlebih dahulu.

```
!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
!curl -fsSL https://deb.nodesource.com/gpgkey/nodesource-repo.gpg.key | sudo gpg --dearmor -o /etc/apt/keyrings/nodesource.gpg

!NODE_MAJOR=20 && echo "deb [signed-by=/etc/apt/keyrings/nodesource.gpg] https://deb.nodesource.com/node_$NODE_MAJOR.x nodistro main"

!sudo apt-get update
!sudo apt-get install nodejs -y

!node -v
```

Gambar 3.3 Kode Program *Install Node JS*

3.2.1.3 Pengambilan Data

Hasil pengumpulan data disimpan pada file dengan format *CSV*. Didapatkan data *tweet* yang memiliki variasi bentuk dengan terdapat *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* sehingga perlu dilakukan pengolahan data lebih lanjut.

```
filename = 'DataPrabowo.csv'
search_keyword = 'prabowo until:2024-04-30 since:2024-04-24 lang:id'
limit = 10000

!npx -y tweet-harvest@2.6.0 -o "{filename}" -s "{search_keyword}" --tab "LATEST" -l {limit} --token {twitter_auth_token}
```

Gambar 3.4 Kode Program Pengambilan Data

3.2.2 Prapemrosesan Data.

Prapemrosesan data menjadi langkah krusial dalam pengolahan data teks. Proses ini melibatkan serangkaian langkah untuk membersihkan dan memperbaiki data teks yang masih mentah, sehingga data menjadi lebih terstruktur dan siap digunakan untuk analisis. Sebelum melakukan prapemrosesan data, langkah awal yang penting adalah mengimpor pustaka yang diperlukan.

```
import numpy as np
import pandas as pd
from pandas import DataFrame
import nltk, emoji, re, string

nltk.download('punkt')
nltk.download('stopwords')

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
```

Gambar 3.5 Pustaka Prapemrosesan data

Tahapan-tahapan dalam prapemrosesan data:

3.2.2.1 Pembersihan Teks

Pembersihan teks yaitu proses menghilangkan karakter khusus seperti tanda baca, simbol, dan format teks yang tidak memiliki dampak signifikan terhadap makna teks.

```
#Untuk menghapus mention
def mention_remove(text):
  return re.sub(r'(?:^|\s)@[\w\d_]+', '', text)

#Untuk menghapus ref character HTML
def htmlChar_remove(text):
  return re.sub(r'&[a-z]+;', ' ', text)

#Untuk menghapus URL (Uniform Resource Locator).
def url_remove(text):
  return re.sub(r'https://\S+',' ',text)

#Untuk menghapus angka
def number_remove(text):
  return re.sub(r'\d+', ' ', text)

#Untuk menghapus emoji
def emoji_remove(text):
  pattern = r"(?:[:=][oO\.]+[:=])|(?:[-;_\*]{2,})"
  newText = re.sub(pattern, '', text) #remove emoji dg re
  return emoji.replace_emoji(newText, replace='') #remove emoji dg lib emoji

#Untuk menghapus # (hastag)
def hastag_remove(text):
  return re.sub(r'#(\w+)', ' ', text)

#Untuk menghapus tanda baca
def punc_remove(text):
  return re.sub(r'[^\w\s]',' ',text)

#ASCII (American Standard Code for Information Interchange) adalah format pengkodean karakter yang paling umum untuk data teks di komputer dan internet.
def ensure_ascii(text):
  return ''.join([x for x in text if x.isascii()])

#Untuk menghapus lebih dari satu karakter kosong (spasi) yang terletak sebelum dan sesudah kata.
def space_remove(text):
  res = re.sub(' +', ' ',text)
  return res.strip()
```

Gambar 3.6 Kode Program Pembersihan Teks

3.2.2.2 Pengubahan Huruf

Pengubahan huruf yaitu proses mengubah teks dari bentuk aslinya menjadi bentuk standar, seperti huruf kecil (*lowercase*).

```
def case_folding():
  lowerWord = data['cleaned_text'].str.lower()
  return lowerWord

lowerTweet = case_folding()

data['case_folded'] = lowerTweet
```

Gambar 3.7 Kode Program Pengubahan Huruf

3.2.2.3 Tokenisasi

Tokenisasi yaitu proses mengidentifikasi batas-batas kata dalam teks, memisahkannya menjadi unit-unit individual untuk analisis lebih lanjut.

```
tokenizedWord = []

def tokenizing(texts):
  for text in texts:
    tokenizedWord.append(nltk.tokenize.word_tokenize(text))

tokenizing(data['case_folded'])
```

Gambar 3.8 Kode Program Tokenisasi

3.2.2.4 Normalisasi Kata

Proses ini bertujuan untuk menyeragamkan terminologi dan penggunaan tata bahasa suatu teks atau dokumen. Hal ini dilakukan dengan mengoreksi kesalahan penulisan, seperti ejaan. Selain itu, proses ini juga bertujuan untuk memastikan konsistensi penggunaan istilah dan singkatan.

```
normWordsDict = {
  'aq': 'aku', 'ats': 'atas', 'aja': 'saja', 'aj': 'saja', 'aje': 'saja', 'ama': 'sama', 'anjlok': 'turun', 'adalh':'a
  'bwt': 'buat', 'brsih': 'bersih', 'bnyk': 'banyak', 'buwat': 'buat', 'bkan':'bukan', 'bkn':'bukan', 'bgni':'begini',
  'blm':'belum', 'byk':'banyak', 'berbecara':'berbicara', 'bgst':'bangsat', 'bisax':'bisa', 'blon':'belum', 'bukane':'
  'cm':'cuma', 'cman':'cuma', 'cairrr':'cair', 'cilaka':'cilaka',
  'dpt':'dapat', 'dapet':'dapat', 'dri':'dari', 'daei':'dari', 'dar':'dari', 'dl':'dulu', 'dng':'dengan', 'dg':'dengar
  'elo':'kamu', 'elu':'kamu', 'ente':'kamu', 'emank':'memang', 'emg':'memang', 'mank':'memang',
  'engga':'tidak',
  'fullll':'penuh',
  'ga':'tidak', 'gak':'tidak', 'gk':'tidak', 'gx':'tidak', 'gw': 'aku', 'guwe': 'aku', 'gue': 'aku', 'gmn': 'gimana',
  'hqq':'hakiki', 'hellooww':'halo',
  'inget':'ingat', 'indo':'indonesia',
  'jngn':'jangan', 'jangn':'jangan', 'jd':'jadi', 'jlas':'jelas', 'jdi':'jadi', 'jng':'jangan', 'jgn':'jangan', 'jkw':
  'krn': 'karena', 'krna': 'karena', 'kyk': 'seperti', 'kbrny': 'kabarnya', 'kagak': 'tidak', 'kaga': 'tidak', 'karna'
  'keluargany' : 'keluarganya', 'kasi' : 'kasih', 'klu' : 'kalau', 'koq' : 'kok', 'knp' : 'kenapa', 'kocaaaakkk' : 'lu
  'lg': 'lagi', 'lu':'kamu', 'loe':'kamu', 'koe':'kamu', 'lo':'kamu', 'laen':'lain', 'luh':'kamu', 'luuuuuuuu':'kamu',
  'mntri':'menteri', 'mendaparkan':'mendapatkan', 'menila':'menilai', 'msuh':'musuh', 'mnyesal':'menyesal', 'moga':'se
  'menjilay':'menjilat', 'mnx':'memang', 'mnjaga':'menjaga', 'menlanggar':'melanggar', 'mkan':'makan', 'memuaskn':'mem
  'nggk':'tidak', 'ndak':'tidak', 'nggak':'tidak', 'ngga':'tidak', 'ngak':'tidak', 'ngk':'tidak', 'ndk':'tidak', 'ngar
  'org':'orang', 'ornag':'orang', 'ora':'tidak', 'ogah':'tidak', 'ojo':'jangan',
  'plg': 'paling', 'pd': 'pada', 'pj': 'puji', 'plosok': 'pelosok', 'pecaya': 'percaya', 'pntes': 'pantas', 'pnjlat':'
  'rkyt': 'rakyat', 'raktyat': 'rakyat', 'rp': 'rupiah', 'rezzziimmm': 'rezim', 'rosak':'rusak', 'rekom':'rekomendasi'
  'samp':'sampai', 'slmpe':'sampai', 'sampe':'sampai', 'sampee':'sampai', 'spt': 'seperti', 'spti': 'seperti', 'silaka
  'sndri':'sendiri', 'slh':'salah', 'swbetulnya':'sebetulnya', 'samapai':'sampai', 'supiir':'supir', 'sgitu':'segitu',
  'trus': 'terus', 'teruuuuus': 'terus', 'trplhnya':'terpilihnya', 'tdk':'tidak', 'terhdp':'terhadap', 'terhsdap':'ter
  'utk': 'untuk', 'udah': 'sudah', 'uda': 'sudah',
  'wes': 'sudah', 'wkl': 'wakil', 'wae': 'saja', 'wa': 'whatsapp', 'wani': 'berani', 'wowo': 'prabowo', 'wedhus': 'kam
  'yg': 'yang', 'yen': 'kalau', 'yo': 'ya',
}

def normalize_word(document):
  return [normWordsDict[term] if term in normWordsDict else term for term in document]

normalizedWords = []
for document in data['tokenized_word']:
  normalizedWords.append(normalize_word(document))
```

Gambar 3.9 Kode Program Normalisasi Kata

3.2.2.5 Penghapusan Kata Henti

Penghapusan kata henti merupakan sebuah teknik pengolahan teks yang bertujuan untuk memurnikan teks dengan menghilangkan kata-kata yang dianggap kurang informatif. Kata-kata yang dimaksud umumnya berupa kata-kata fungsional seperti "yang", "dan", "di", "dari", dan lain sebagainya.

```
stopFactory = StopWordRemoverFactory().get_stop_words()
moreStopword = ['anjing', 'koar', 'jngn', 'wkwk','rmh', 'bbrp', 'thn', 'oalaaaaaah', 'lonte',
                'omon', 'dr', 'dollar', 'yad', 'etc', 'ruarrr', 'si', 'yv', 'ps', 'gp', 'ab', 'ntah',
                'allah', 'admin', 'tolol', 'uud', 'ber', 'dlm', 'tp', 'hy', 'pilgubsu', 'ad', 'mpe',
                'cuuuk', 'nepotisme', 'wkwkwkwkw', 'wkwkwkwwkwk' 'bet', 'ni', 'nih', 'move', 'hadeuhhh', 'aja', 'kalo', 'menggonggong', 'secuil',
                'tulang', 'kacung', 'tidur', 'yah', 'hahaha', 'gak', 'amnesia', 'lha', 'dn', 'prab', 'swt', 'jiakkhh', 'bangs', 'awowokwokwokwok', 'aka',
                'bahasu', 'amu', 'gothal', 'gothel', 'cc', 'worldhoaksorganization', 'wef', 'bacot', 'kontol', 'tdk', 'bs', 'lbh', 'jg', 'kontollll',
                'sy', 'adalaaaahhh', 'jirr', 'kratingdeng', 'penjilmek', 'oo', 'bhuawkwkwkwk', 'bokep', 'kalqu', 'th', 'bklan', 'gacor', 'hhh', 'wek',
                'ogaaaahhhhh', 'tai', 'ajaa', 'tuuhhh', 'lc', 'nt', 'please', 'yam', 'kek', 'asssuuuuuu', 'terhdp', 'su', 'wkwkw','yntkts', 'wkwkkw', 'wkwkwkwk',
                'wkwkwkwkwkww', 'pnyblebih', 'guoblokkkkk', 'tertoolol', 'hxhx', 'wkkkkkkkkk', 'wkk', 'cooookkkkkkkk', 'wkwkwkww', 'wkwkwkwkwkwkwkwkwkwkwkw'
                ]
englishLangStopwords = list(stopwords.words('english'))
allStopWords = stopFactory + moreStopword + englishLangStopwords

filtered_sentences = []

for words in data['normalized_word']:
  filteredWord=[word for word in words if word not in allStopWords]
  filtered_sentences.append(filteredWord)
```

Gambar 3.10 Kode Program Penghapusan Kata Henti

3.2.2.6 Stemming

Stemming merupakan proses penghilangan imbuhan atau akhiran kata untuk mendapatkan bentuk dasarnya. Proses ini bertujuan untuk mereduksi kata menjadi bentuk yang lebih ringkas dan mudah dianalisis.

```
factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stemming_word(text):
  return stemmer.stem(text)

stemmedWords = []
for document in filtered_sentences:
  stemmedWords.append([stemming_word(term) for term in document])
```

Gambar 3.11 Kode Program *Stemming*

3.2.3 Pelabelan Data

Proses pelabelan merupakan kegiatan pemberian label kepada kata atau kalimat yang terdapat dalam suatu dokumen. Tujuannya adalah untuk memudahkan analisis lebih lanjut terkait sifat positif, netral atau negatif dari konten tersebut. Penelitian ini menggunakan pustaka *TextBlob* untuk melakukan pelabelan data secara otomatis.

```
import googletrans
from googletrans import Translator

import textblob
from textblob import TextBlob
```

Gambar 3.12 Pustaka Pelabelan Data

Tahapan pelabelan data adalah sebagai berikut.

3.2.3.1 Menerjemahkan Teks

Merupakan proses mengubah teks dari bahasa Indonesia ke bahasa Inggris dengan mempertahankan makna aslinya.

```
translator = Translator()

def clean_and_translate(sentence):
    return space_remove(punc_remove(translator.translate(sentence, dest='en').text)).lower()

df['translated_text'] = df['preprocessed_data'].apply(clean_and_translate)
```

Gambar 3.13 Kode Program Menerjemahkan Teks

3.2.3.2 Pelabelan Data

Dalam penelitian akan dilakukan pelabelan data secara otomatis dengan ketentuan -1 sebagai label negatif, 0 sebagai label netral, dan 1 sebagai label positif.

```
numberLabel = []
textLabel = []
polaritys = []
for sentence in df['translated_text']:
  blob = TextBlob(sentence)
  polarity = blob.sentiment.polarity
  subjectivity = blob.sentiment.subjectivity
  if polarity > 0:
    sentimentLabel = "Positif"
    numLabel = 1
  elif polarity < 0:
    sentimentLabel = "Negatif"
    numLabel = -1
  else:
    sentimentLabel = "Netral"
    numLabel = 0

  numberLabel.append(numLabel)
  textLabel.append(sentimentLabel)
  polaritys.append(polarity)

df['label_text'] = textLabel
df['label'] = numberLabel
df['polarity'] = polaritys
```

Gambar 3.14 Kode Program Pelabelan Data Dengan *TextBlob*

preprocessed_data	translated_text	label_text	label	polarity
persis waktu sby menang pilpres ngambek	just when sby won the presidential election he...	Netral	0	0.000000
prabowo anies segani hormat dukung sosmed anti...	prabowo anies respectfully supports anti anies...	Positif	1	0.177778
suka suka capres kalah	like it or not the presidential candidate loses	Negatif	-1	-0.300000
selamat garut	good luck garut	Positif	1	0.700000
selamat prabowo gibran moga bawa indonesia beb...	congratulations prabowo gibran may indonesia b...	Positif	1	0.400000
gua wakil gibran	i m gibran s representative	Netral	0	0.000000
jendral	general	Positif	1	0.050000
bangga curang hebat	proud cheat great	Positif	1	0.800000
rambut putih dateng	white hair comes	Netral	0	0.000000
bayar utang bro	pay the debt bro	Netral	0	0.000000

Gambar 3.15 Contoh Hasil Pelabelan Data

Proses pelabelan data *tweet* menghasilkan 5.000 data latih yang terbagi menjadi 1.781 data positif, 2.361 data netral, dan 858 data negatif.

3.2.4 Pelatihan Data

Pelatihan data merupakan kumpulan data yang digunakan untuk melatih model dalam analisis sentimen. Dalam penelitian ini digunakan untuk melatih model *LR*. Proses pelatihan data dimulai dengan ekstraksi fitur pada data teks menggunakan metode *TF-IDF*. Selanjutnya, dilakukan pelatihan data untuk membangun model klasifikasi yang dapat digunakan untuk melakukan klasifikasi sentimen secara otomatis. Implementasi perhitungan *TF-IDF* memanfaatkan pustaka *Sklearn* dan *TfidfVectorizer* dalam bahasa pemrograman *Python* untuk menghasilkan nilai perhitungan secara otomatis.

```
(0, 4420)     0.5357597851128865
(0, 5151)     0.3095561285012576
(0, 3969)     0.2461099983897175
(0, 5788)     0.4690580766259601
(0, 6999)     0.35387536339598985
(0, 5107)     0.4596988933954824
(1, 5251)     0.26267798188834585
(1, 279)      0.39114393962289445
(1, 6226)     0.4467211125576177
(1, 1587)     0.22859398580889062
(1, 2328)     0.3216084557218544
(1, 5825)     0.43446164580794777
(1, 257)      0.445403871152612
(1, 5286)     0.18872785874628265
(2, 2828)     0.363071607512449
(2, 1052)     0.4391589138679138
(2, 6314)     0.8217776196688984
(3, 1900)     0.932841562371551
(3, 5868)     0.36028685725710796
(4, 5892)     0.4015649819646854
(4, 3289)     0.39806187535001486
(4, 597)      0.46064050599006
(4, 2443)     0.25353092289793233
(4, 578)      0.3870479695105656
(4, 4148)     0.2704891935554551
```

Gambar 3.16 Hasil Perhitungan TF-IDF Awal

```
(4997, 4471)  0.36999852453316956
(4997, 1714)  0.36999852453316956
(4997, 393)   0.3529974694962735
(4997, 2297)  0.5249016225004505
(4997, 4165)  0.5710282452840465
(4998, 2296)  0.3064389263379331
(4998, 1321)  0.4347322461218552
(4998, 3066)  0.2301360738222941
(4998, 6694)  0.1946572678472767
(4998, 4971)  0.2160555175317203
(4998, 5477)  0.44956578953320714
(4998, 5148)  0.27534517352366217
(4998, 4201)  0.20240635715811353
(4998, 5092)  0.29586630048777035
(4998, 1052)  0.41610027958049517
(4999, 4641)  0.40325260535312746
(4999, 1163)  0.4175198708157768
(4999, 385)   0.3001250651614124
(4999, 6880)  0.32597819368955566
(4999, 3091)  0.3165030263366099
(4999, 4233)  0.3345009118919841
(4999, 6160)  0.3165030263366099
(4999, 3850)  0.18177096724609787
(4999, 3780)  0.2599079026255676
(4999, 1674)  0.23215439356774278
```

Gambar 3.17 Hasil Perhitungan *TF-IDF* Akhir

Berdasarkan hasil perhitungan *TF-IDF*, dihasilkan perhitungan data latih dengan detail akurasi dan 10 nilai positif yang terlihat pada Gambar 3.17.

```
auc (train data) : 0.9488
top 10 positive scores:
 [2.01270649e-01 4.43017469e-01 6.59419186e-01 4.95822393e-01
 1.08847004e-01 8.76593431e-01 1.21172932e-01 1.98806101e-04
 7.48734170e-01 8.25377899e-01]
top 10 neutral scores:
 [1.14817598e-01 1.19312331e-01 1.86964937e-01 2.31773860e-02
 1.09155036e-02 9.40573329e-02 1.59261088e-02 2.37042784e-05
 9.84082966e-02 1.37576641e-01]
top 10 negative scores:
 [0.68391175 0.4376702  0.15361588 0.48100022 0.88023749 0.02934924
 0.86290096 0.99977749 0.15285753 0.03704546]
```

Gambar 3.18 Hasil Perhitungan Data Latih

3.2.5 Model *Logistic Regression*

Berikutnya, setelah perhitungan *TF-IDF* dilakukan, model klasifikasi *LR* dibuat dengan menggunakan variabel X dan y pada data latih yang telah disiapkan. Pembuatan model ini dikemas dalam sebuah fungsi untuk memudahkan pemanggilan dan eksekusi pada tahap selanjutnya.

```
modelLR = LogisticRegression()

parameters = {
  'solver': ['saga'],
  'penalty':['l1'],
  # 'solver': ['saga', 'sag'],
  # 'penalty':[None, 'elasticnet', 'l1', 'l2'],
  # 'penalty':[None],
  'C':np.arange(1.0, 2.0, 0.1)
  # 'C':[1, 10, 100, 1000]
}

grid_search = GridSearchCV(estimator = modelLR, param_grid = parameters, scoring = 'accuracy', cv = 5, verbose=0)

grid_search.fit(X_train_tfidf, Y_train)

print('GridSearch CV best score : {:.4f}\n\n'.format(grid_search.best_score_))
print('Parameters that give the best results :','\n\n', (grid_search.best_params_))
print('\n\nEstimator that was chosen by the search :', '\n\n', (grid_search.best_estimator_))
```

Gambar 3.19 Kode Program Untuk Mencari Parameter Terbaik Model

```
X_train,X_test,y_train,y_test = train_test_split(dfLabelled['preprocessed_data'],dfLabelled['label'],stratify=dfLabelled['label'], test_size=0.2,random_state=42)

pipeline = Pipeline([
  ('bow',CountVectorizer()),
  ('tfidf',TfidfTransformer()),
  ('classifier',LogisticRegression(solver='saga', penalty='l1', C=1.9000000000000008))
  ])

X_train = np.asarray(X)
model = pipeline.fit(X_train, np.asarray(y))
```

Gambar 3.20 Kode Program Pembuatan Model Klasifikasi

Selanjutnya model klasifikasi digunakan untuk menganalisis 35.587 data *tweet*. Data tersebut terdiri dari 5.000 *tweet* yang telah diberi label dan 30.587 *tweet* yang belum diberi label. Model klasifikasi yang telah dibuat dieksekusi pada seluruh data *tweet* untuk memprediksi sentimen positif dan negatif dari setiap *tweet*. Hasil prediksi untuk 35.587 *tweet* tersebut disajikan dalam Gambar 3.21.

	username	preprocessed_data	label_text	label
0	UbaiBaiquni	persis waktu sby menang pilpres ngambek	Netral	0
1	F5Fahry	prabowo anies segani hormat dukung sosmed anti...	Positif	1
2	holderdiamond_	suka suka capres kalah	Negatif	-1
3	SubzerokillZ	selamat garut	Positif	1
4	MasWahyuOk	selamat prabowo gibran moga bawa indonesia beb...	Positif	1
...	...	...	...	...
30582	rachman0565	haram pajang gambar cundang curang	Netral	0
30583	edykhumaedi1	aaassyyuuuudahlah kauu	Netral	0
30584	AtokSsk	maaf asam sulfate puja puji usak konstitusi ba...	Negatif	-1
30585	Erwan86134286	amoga anies sllu konsisten lbih pilih istiraha...	Positif	1
30586	radaraktual	nilai suka bikin narasi adu domba partai gelor...	Netral	0

35587 rows × 4 columns

Gambar 3.21 Hasil Data Prediksi

Gambaran data positif, netral, dan negatif dapat dilihat pada Gambar 3.22.

label_text	username	preprocessed_data	label
Negatif	3337	3337	3337
Netral	22050	22050	22050
Positif	10200	10200	10200

Gambar 3.22 Gambaran Data Positif, Netral, dan Negatif

3.2.6 Pengujian

Pengujian merupakan proses untuk mengevaluasi sejauh mana model yang telah dilatih dapat menghasilkan prediksi yang akurat terhadap label atau kelas data uji. Pada proses pengujian dilakukan pengujian pada 5.000 data *tweet* yang sudah dilabeli dengan pustaka *TextBlob*. Data *tweet* yang sudah dilakukan prediksi dilakukan pengujian dengan hasil pada Gambar 3.23 dan Gambar 3.24.

```
model = LogisticRegression(solver='saga', C=1.9000000000000008, penalty='l1')
model.fit(X_train_tfidf, Y_train)
y_pred = model.predict(X_test_tfidf)
accuracy = accuracy_score(Y_test, y_pred)
print(f'Accuracy: {accuracy} / {accuracy * 100:.2f}%')
```

```
Accuracy: 0.9346726608597921 / 93.47%
```

Gambar 3.23 Hasil Akurasi Model

```
print('Training set score: {:.4f}'.format(model.score(X_train_tfidf, Y_train)))
print('Test set score: {:.4f}'.format(model.score(X_test_tfidf, Y_test)))
```

```
Training set score: 0.9553
Test set score: 0.9347
```

Gambar 3.24 Hasil Perhitungan Akurasi Data Latih dan Data Uji

3.2.7 Evaluasi Model

Confusion matrix dimanfaatkan untuk mengevaluasi model melalui berbagai metode untuk menentukan persentase prediksi yang benar saat pengujian.

```
from sklearn.metrics import confusion_matrix
conf_matrix = confusion_matrix(Y_test, y_pred)
print(f'\nConfusion Matrix:\n {conf_matrix}')
```

Gambar 3.25 Kode Program Perhitungan *Confusion Matrix*

```
import seaborn as sns
cm_matrix = pd.DataFrame(
  data=conf_matrix, columns=['Actual Negative: -1', 'Actual Neutral:0', 'Actual Positive: 1'],
  index=['Predict Negative:-1', 'Predict Neutral:0', 'Predict Positive: 1']
)

sns.heatmap(cm_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

Gambar 3.26 Kode Visualisasi Hasil *Confusion Matrix*

Metode evaluasi yang digunakan:

3.2.7.1 *Accuracy*(akurasi)

Akurasi didefinisikan sebagai proporsi prediksi yang tepat (positif benar dan negatif benar) terhadap total prediksi yang dilakukan. Nilai akurasi mencerminkan frekuensi model dalam menghasilkan prediksi yang benar.

Untuk mengukur tingkat akurasi, menggunakan persamaan (2).

$$Accuracy = \frac{(\mathrm{TP} + \mathrm{TN})}{(\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN})} \quad (2)$$

3.2.7.2 *Precision*(presisi)

Presisi adalah proporsi dari hasil positif yang diprediksi secara benar terhadap total hasil positif yang diprediksi oleh model. Persamaan (3) digunakan untuk menghitung nilai presisi.

$$Precision = \frac{(\mathrm{TP})}{(\mathrm{TP} + \mathrm{FP})} \quad (3)$$

3.2.7.3 *Recall*(sensivitas)

Sensitivitas adalah proporsi dari hasil positif yang diprediksi secara benar terhadap total hasil positif yang sebenarnya. Persamaan (4) digunakan untuk menghitung nilai sensivitas.

$$Recall = \frac{(\mathrm{TP})}{(\mathrm{TP} + \mathrm{FP})} \quad (4)$$

3.2.7.4 *F1-Score*

Merupakan metrik evaluasi model klasifikasi yang dihitung sebagai rata-rata harmonik antara presisi dan *recall*. Metrik ini menggabungkan kedua metrik tersebut untuk memberikan gambaran yang komprehensif tentang kinerja model. Persamaan (5) digunakan untuk menghitung nilai sensivitas.

$$F1 - Score = \frac{(2 \text{ x } Recall \text{ x } Precision)}{(Recall + Precision)} \quad (5)$$

```
from sklearn.metrics import classification_report
report = classification_report(Y_test, y_pred, target_names=["Negatif", "Netral", "Positif"])
print("\nClassification Report:\n", report)
```

Gambar 3.27 Kode Program Perhitungan *Classification Report*