BAB 4
HASIL PENELITIAN

4.1 RINGKASAN HASIL PENELITIAN

Penelitian ini menggunakan media sosial *X* sebagai sumber data utama, dengan mengumpulkan informasi melalui beberapa kata kunci seperti *"jersey Erspo", "jersey Timnas Indonesia", "launching Erspo",* dan lain-lain. Data yang diperoleh melalui teknik web scraping kemudian diolah dan dibersihkan untuk menghilangkan duplikasi, entri yang tidak relevan, dan spam. Setelah proses pembersihan, data tersebut diberi label sesuai dengan kategorinya, apakah menunjukkan sentimen positif atau negatif. Pemberian label ini dilakukan melalui bantuan algoritma praproses yang mengidentifikasi konteks dan nuansa dari setiap entri teks. Data yang telah diberi label ini kemudian diberikan pembobotan untuk menentukan signifikansi masing-masing kata atau frasa dalam konteks sentimen yang diekspresikan. Dengan data yang sudah terstruktur dan diberi bobot, proses klasifikasi dilakukan menggunakan metode *Naive Bayes,* yang dikenal efektif untuk tugas-tugas klasifikasi teks. Model klasifikasi yang dihasilkan kemudian dievaluasi untuk memastikan akurasinya dalam memprediksi sentimen dari data baru yang masuk, sehingga memberikan wawasan yang berguna mengenai persepsi publik terhadap jersey Timnas Indonesia produksi *Erspo*.

4.2 HASIL PENELITIAN

Proses penelitian ini mencakup beberapa tahapan yang berlangsung dari Februari 2024 hingga Juni 2024. Data dikumpulkan melalui proses crawling pada media sosial *X* dengan menggunakan kata kunci terkait *jersey* terbaru Tim Nasional Indonesia yang dibuat oleh *apparel Erspo*. Penjelasan lebih rinci dapat dilihat di bawah ini:

4.2.1 *Crawling Data*

Pengambilan data adalah tahap pengumpulan data dari media sosial *X* untuk mendapatkan tweet tentang *jersey Erspo*. Proses ini dilakukan

menggunakan perangkat lunak *Google Colab*, dan data yang dikumpulkan kemudian disimpan dalam format CSV. Pengumpulan tweet dilakukan dengan bantuan perangkat lunak *Tweet Harvest* dan berhasil mengumpulkan sebanyak 2158 data tweet, yang melibatkan beberapa tahapan berikut:

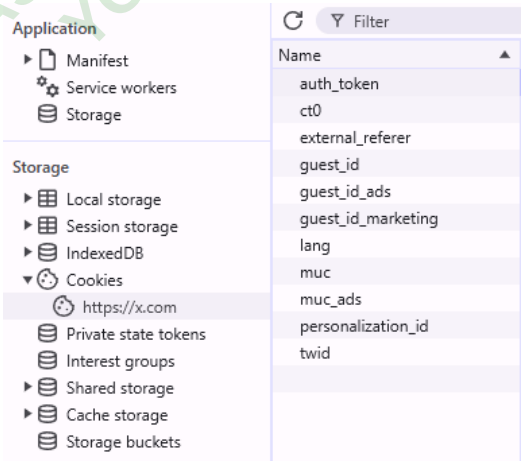
4.2.1.1 Registrasi

Memiliki akun *X* diperlukan sebelum memulai proses pengambilan data. Registrasi akun *X* dapat dilakukan melalui situs web atau aplikasi seluler *X*.

4.2.1.2 *Copy Auth Token*

Pengguna dapat melihat auth token cookie sebagai kode akses pada akun masing-masing. Auth token tersebut bersifat pribadi dan tidak boleh disebarluaskan demi keamanan data pengguna. Berikut adalah cara untuk mendapatkan auth token:

1. Masuk ke akun *X* melalui situs web *https://x.com.*
2. Aktifkan fitur inspect element, lalu pilih panel Application.
3. Pilih menu *Cookie,* kemudian salin kode pada bagian *auth_token*. Atau, bisa dilihat dari Gambar 4.1.



Gambar 4.1 *Auth Token*

4.2.1.3 *Install Node.js*

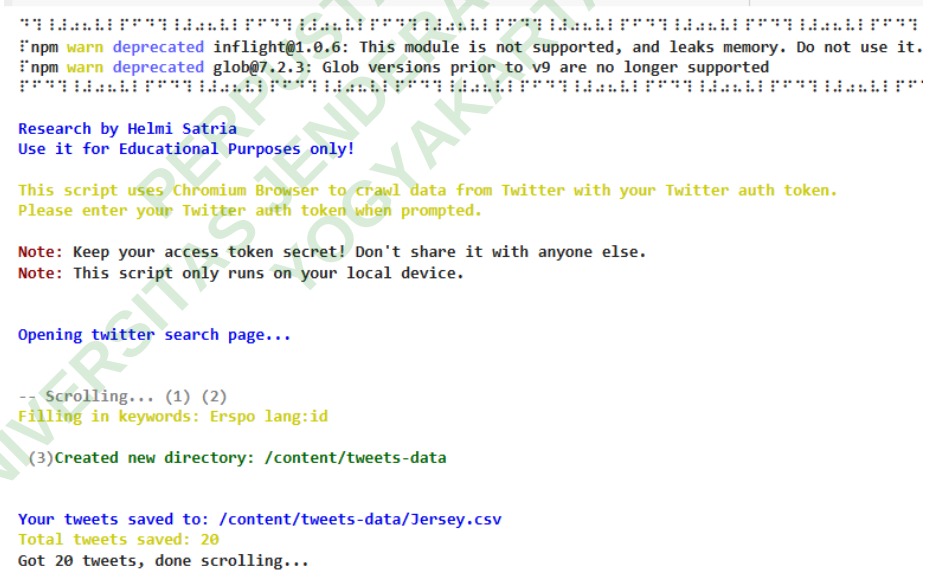
Node.js digunakan untuk menjalankan program *Tweet Harvest. Node.js* dapat diunduh melalui situs web resminya di *https://nodejs.org/en.*

4.2.1.4 *Install Tweet Harvest*

Setelah *Node.js* berhasil dipasang, langkah berikutnya adalah menginstal *Tweet Harvest*. *Tweet Harvest* dapat dijalankan dengan memasukkan kode `npx tweet-harvest@latestversion`. Dalam penelitian ini, penulis menggunakan versi 2.6.1.

4.2.1.5 Mengambil Data *Tweet*

Tweet Harvest akan membuka instance Chromium Browser dan menavigasi ke halaman pencarian di media sosial *X*. Program akan memasukkan parameter pencarian dan mulai melakukan crawling pada tweet yang dihasilkan secara bertahap. Data yang terkumpul akan disimpan dalam format CSV. Dalam penelitian ini, penulis melakukan crawling data pada periode 25 Februari 2024 hingga 25 April 2024 dengan kata kunci "*Jersey Erspo*". Gambar 4.2 menunjukkan proses crawling data.



Gambar 4.2 *Instance Chronium Brwoser*

4.2.2 *Preprocessing Data*

Sebelum memproses data teks yang telah diambil (crawling), preprocessing melibatkan beberapa tahap untuk memperbaiki kualitas data teks. Proses ini memerlukan beberapa pustaka Python yang penting. Di antaranya adalah pandas untuk persiapan dan pembersihan data, numpy untuk perhitungan

numerik, nltk untuk pemrosesan bahasa alami pada teks, string yang berisi konstanta dan fungsi untuk manipulasi string, serta re untuk operasi ekspresi reguler. Sebelum melanjutkan preprocessing data, instalasi pustaka-pustaka ini perlu dilakukan, seperti yang ditunjukkan dalam Gambar 4.3.

```
import pandas as pd
import numpy as np
import nltk
import string
import re
```

Gambar 4.3 Modul *Preprocessing*

Berikut merupakan tahapan-tahapan dari *preprocessing* data:

4.2.2.1 *Cleaning Text*

Pada tahap pembersihan (cleaning), dilakukan penghapusan karakter tanda baca, emotikon, angka, tautan, dan elemen lainnya. Proses ini menggunakan pengulangan dengan modul `re`. Rincian proses cleaning dapat dilihat dalam Gambar 4.4.

```
#cleaning
def cleaning(Text):
    Text = re.sub(r'\$\w*', '', Text)
    Text = re.sub('((www\.[^\s]+)|(https?://[^\s]+))', ' ', Text)
    Text = re.sub('&quot;'," ", Text)
    Text = re.sub(r"\d+", " ", str(Text))
    Text = re.sub(r"\b[a-zA-Z]\b", "", str(Text))
    Text = re.sub(r"[^\w\s]", " ", str(Text))
    Text = re.sub(r'(.)\1+', r'\1\1', Text)
    Text = re.sub(r"\s+", " ", str(Text))
    Text = re.sub(r'#', '', Text)
    Text = re.sub(r'[^a-zA-z0-9]', ' ', str(Text))
    Text = re.sub(r'\b\w{1,2}\b', '', Text)
    Text = re.sub(r'\s\s+', ' ', Text)
    Text = re.sub(r'^RT[\s]+', '', Text)
    Text = re.sub(r'^b[\s]+', '', Text)
    Text = re.sub(r'^link[\s]+', '', Text)
    return Text
def remove_emoji(Text):
    emoji = re.compile("["
                       u"\U0001F600-\U0001F64F"
                       u"\U0001F300-\U0001F5FF"
                       u"\U0001F680-\U0001F6FF"
                       u"\U0001F1E0-\U0001F1FF"
                       u"\U00002702-\U000027B0"
                       u"\U000024C2-\U0001F251"
                       "]+", flags=re.UNICODE)
    return emoji.sub(r'', Text)
tweets['cleaning'] = tweets['remove_user'].apply(cleaning)
tweets[['full_text','cleaning']].head()
```

	full_text	cleaning
0	@ezahimovic Garuda Pancasila bebas dipake siap...	Garuda Pancasila bebas dipake siapa aja mas s...
1	Cek JERSEY OLAHRAGA INDONESIA // JERSEY VINTAG...	Cek JERSEY OLAHRAGA INDONESIA JERSEY VINTAGE I...
2	@Jerseyforum jersey fantasy timnas dari appare...	jersey fantasy timnas dari apparel nine jogja...
3	@Jerseyforum Dapat hadiah giveaway @nthm_id su...	Dapat hadiah giveaway suruh desain jersey tim...
4	kalau udah ngikutin timnas dari lama tapi beli...	kalau udah ngikutin timnas dari lama tapi beli...

Gambar 4.4 *Cleaning text*

4.2.2.2 *Case folding*

Case folding berfungsi untuk mengubah semua huruf dalam teks menjadi huruf kecil standar. Kode yang digunakan untuk melakukan *case folding* dapat dilihat pada Gambar 4.5.

```
#case folding - ubah jadi huruf kecil
tweets['case_folding'] = tweets['cleaning'].str.lower()
tweets[['cleaning','case_folding']].head()
```

	cleaning	case_folding
0	Garuda Pancasila bebas dipake siapa aja mas s...	garuda pancasila bebas dipake siapa aja mas s...
1	Cek JERSEY OLAHRAGA INDONESIA JERSEY VINTAGE I...	cek jersey olahraga indonesia jersey vintage i...
2	jersey fantasy timnas dari apparel nine jogja...	jersey fantasy timnas dari apparel nine jogja...
3	Dapat hadiah giveaway suruh desain jersey tim...	dapat hadiah giveaway suruh desain jersey tim...
4	kalau udah ngikutin timnas dari lama tapi beli...	kalau udah ngikutin timnas dari lama tapi beli...

Gambar 4. 5 *Case folding*

4.2.2.3 Tokenisasi

Tokenisasi dilakukan untuk mengubah data menjadi daftar kata-kata atau list. Menggunakan library NLTK dengan mengimpor fungsi `word_tokenize` dari modul `nltk.tokenize`. Fungsi `word_tokenize_wrapper` digunakan sebagai perantara *(wrapper)* yang memfasilitasi penggunaan fungsi `word_tokenize` dari NLTK untuk setiap baris teks dalam kode tersebut. Contoh kode fungsi tokenisasi pada Gambar 4.6.

```
#tokenisasi - membagi kalimat jadi perkata (dipisah)
nltk.download('punkt')
from nltk.tokenize import word_tokenize

def word_tokenize_wrapper(Tweets):
    return word_tokenize(Tweets)

tweets['tokenisasi'] = tweets['case_folding'].apply(lambda x: word_tokenize_wrapper(x.lower()))
tweets[['case_folding','tokenisasi']].head()
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

	case_folding	tokenisasi
0	garuda pancasila bebas dipake siapa aja mas s...	[garuda, pancasila, bebas, dipake, siapa, aja,...
1	cek jersey olahraga indonesia jersey vintage i...	[cek, jersey, olahraga, indonesia, jersey, vin...
2	jersey fantasy timnas dari apparel nine jogja...	[jersey, fantasy, timnas, dari, apparel, nine,...
3	dapat hadiah giveaway suruh desain jersey tim...	[dapat, hadiah, giveaway, suruh, desain, jerse...
4	kalau udah ngikutin timnas dari lama tapi beli...	[kalau, udah, ngikutin, timnas, dari, lama, ta...

Gambar 4.6 Tokenisasi

4.2.2.4 Normalisasi

Normalisasi merupakan proses mengubah kalimat slang menjadi formal dengan menggunakan dan merujuk pada kamus 'colloquial-indonesian-lexicon.csv'. Kamus tersebut berisi daftar kata-kata slang beserta bentuk formalnya dalam bahasa Indonesia. Selanjutnya, dilakukan iterasi atau loop. Dalam setiap iterasi ini, setiap kata dalam teks diperiksa keberadaannya dalam kamus `dict_slang`. Jika kata tersebut terdapat dalam kamus, maka kata slang akan diganti dengan bentuk formalnya sesuai dengan kamus tersebut. Jika kata tersebut tidak ada dalam kamus, kata tersebut akan tetap dipertahankan. Gambar 4.7 menunjukkan contoh kode untuk proses normalisasi:

```
#Normalisasi-menormalisasikan kata yang non formal menjadi formal sesuai
#dengan kamus colloquial-indonesian-lexicon
def normalization (Tweets):
  tweets_slang = pd.read_csv('colloquial-indonesian-lexicon.csv')
  dict_slang ={}
  for i in range(tweets_slang.shape[0]):
    dict_slang[tweets_slang["slang"][i]]=tweets_slang["formal"][i]

  drop_slang = []
  for teks in Tweets:
    normalisasi_teks = [dict_slang[word] if word in dict_slang.keys() else word for word in teks]
    drop_slang.append(normalisasi_teks)

  return drop_slang

tweets['normalisasi'] = normalization(tweets['tokenisasi'])
tweets[['tokenisasi','normalisasi']].head()
```

	tokenisasi	normalisasi
0	[garuda, pancasila, bebas, dipake, siapa, aja,...	[garuda, pancasila, bebas, dipakai, siapa, saj...
1	[cek, jersey, olahraga, indonesia, jersey, vin...	[cek, jersey, olahraga, indonesia, jersey, vin...
2	[jersey, fantasy, timnas, dari, apparel, nine,...	[jersey, fantasy, timnas, dari, apparel, nine,...
3	[dapat, hadiah, giveaway, suruh, desain, jerse...	[dapat, hadiah, giveaway, suruh, desain, jerse...
4	[kalau, udah, ngikutin, timnas, dari, lama, ta...	[kalau, sudah, mengikuti, timnas, dari, lama, ...

Gambar 4.7 Normalisasi

4.2.2.5 *Stopword Removal*

Tahap berikutnya akan dilakukan penghapusan kata-kata yang tidak memiliki makna penting. Digunakan *library* NLTK dengan modul *stopwords* dalam bahasa Indonesia. Dengan menggunakan perintah *from nltk.corpus import stopwords*, kode tersebut mengimpor daftar stopwords (kata-kata

pengisi) dalam bahasa Indonesia dari NLTK. Daftar ini digunakan untuk mengidentifikasi kata-kata yang harus dihapus dari teks. Kemudian, *list_stopwords_id.extend* digunakan untuk menambahkan kata-kata tambahan yang tidak memiliki makna selain yang sudah disediakan dalam corpus stopwords NLTK. Gambar 4.8 menunjukkan kode proses penghapusan stopwords:

```
#Stopword removal - filtering/menghapus kata yang tidak ada didalam kamus corpus nltk
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
# Bahasa Indonesia stopwords
list_stopwords_id = stopwords.words('indonesian')
list_stopwords_id.extend(['yg', 'dg', 'rt', 'dgn', 'ny', 'gt', 'klo',
                       'kalo', 'amp', 'xbf', 'xad', 'xef',
                       'xe', 'ga', 'krn', 'nya', 'nih', 'sih',
                       'si', 'xc', 'tdk', 'tuh', 'utk', 'ya',
                       'jd', 'jgn', 'sdh', 'xae', 'n', 't',
                       'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt',
                       '&', 'yah', 'no', 'je', 'xbb', 'xb', 'sch',
                       'injirrr', 'ah', 'oena', 'bu', 'eh', 'xac', 'xbc', 'xf', 'xa'])
list_stopwords = set(list_stopwords_id)
def stopwords_removal(Text):
    return [word for word in Text if word not in list_stopwords]

tweets['stopword_removal'] = tweets['normalisasi'].apply(stopwords_removal)
tweets[['normalisasi','stopword_removal']].head()
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

	normalisasi	stopword_removal
0	[garuda, pancasila, bebas, dipakai, siapa, saj...	[garuda, pancasila, bebas, dipakai, mas, jerse...
1	[cek, jersey, olahraga, indonesia, jersey, vin...	[cek, jersey, olahraga, indonesia, jersey, vin...
2	[jersey, fantasy, timnas, dari, apparel, nine,...	[jersey, fantasy, timnas, apparel, nine, jogja...
3	[dapat, hadiah, giveaway, suruh, desain, jerse...	[hadiah, giveaway, suruh, desain, jersey, timn...
4	[kalau, sudah, mengikuti, timnas, dari, lama, ...	[mengikuti, timnas, beli, jersey, gapantes, di...

Gambar 4.8 *Stopword removal*

4.2.2.6 *Stemming*

Proses *stemming* dalam penelitian menggunakan pustaka Sastrawi dalam bahasa Indonesia untuk menghilangkan imbuhan kata. Modul `*StemmerFactory*` digunakan untuk membuat objek stemmer. Sementara `*StopWordRemoverFactory*` digunakan untuk membuat objek remover stop words yang akan digunakan untuk mengurangi noise seperti "dan", "atau", "di", "dari", dan sejenisnya. Penggunaan modul `swifter` bertujuan untuk mempercepat proses stemming. Gambar 4.9 merupakan kode dari proses *stemming.*

```
!pip install swifter
!pip install Sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
import swifter
#buat stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()
#stemmed wrapper
def stemmed_wrapper(term):
  return stemmer.stem(term)
term_dict = {}
for Tweets in tweets['stopword_removal']:
  for term in Tweets:
    if term not in term_dict:
      term_dict[term] = ' '
for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
#memmulai stemming
def apply_stemmed_term(Tweets):
  return [term_dict[term] for term in Tweets]
tweets['stemming'] = tweets['stopword_removal'].swifter.apply(apply_stemmed_term)
tweets[['stopword_removal','stemming']].head()
```

Gambar 4.9 *Stemming*

Hasil dari proses preprocessing tersebut menghasilkan sebuah kolom baru yang disebut *"clean_text"*, merupakan representasi dari data yang telah dibersihkan dan siap untuk dilakukan pelabelan. Detail lengkap mengenai hasil dari *stemming* dapat dilihat pada Tabel 4.1 yang disajikan di bawah.

Tabel 4.1 Hasil *Stemming*

No	clean_text
1.	garuda pancasila bebas pakai mas *jersey* fantasy cantum logo kreasi *jersey* pakai timnas legitimasi *jersey* curi pasar timnas
2.	ikut timnas beli *jersey* gapantes bilang fomo
3.	gue *jersey* suka timnas beli orisinal suka sayang duit gue kpopers heran lihat laku fans fomo lebih fans kpop tau
4.	putih eyy kesel perkara *jersey* punaynya timnas jelek
5.	*jersey* indonesia timnas klub font selera aneh aneh bentuk font gampang baca huruf angka

4.2.3 Labeling Data

Penulis menggunakan proses pelabelan manual untuk analisis sentimen, penulis terlebih dahulu mengumpulkan data teks. Setiap entri teks kemudian dibaca secara cermat dan diberi label berdasarkan sentimen yang diungkapkan,

apakah positif, atau negatif. Proses tersebut melibatkan pemahaman kontekstual dan subjektivitas untuk memastikan bahwa label yang diberikan akurat dan konsisten dengan maksud sebenarnya dari penulis teks. Hasil pelabelan manual kemudian digunakan sebagai data latih untuk model analisis sentimen otomatis. Contoh data yang telah diberi label secara manual dapat dilihat pada Tabel 4.2 berikut:

Tabel 4.2 *Labeling* Data

No	Kelas	Label	*clean_text*
1	Positif	1	ikut timnas beli *jersey* gapantes bilang fomo
2	Positif	1	*jersey* timnas indonesia laris *jersey* timnas indonesia
3	Positif	1	timnas indonesia luncur *jersey* spesial hut indonesia september
4	Positif	1	*jersey* kiper kuning kerah hitam launching erspo
5	Positif	1	*jersey* keren timnas timnasday
6	Negatif	0	putih eyy kesel perkara *jersey* pnyny timnas jelek
7	Negatif	0	launching gimmick jelek lag
8	Negatif	0	kemahalan *jersey* baru timnas
9	Negatif	0	boikoterspo desainer banyak bacot
10	Negatif	0	erspo desain jelek payah euyy

Dari Tabel 4.2, dapat diperoleh informasi bahwa kelas positif diberi label dengan nilai 1, sementara kelas negatif diberi label dengan nilai 0. Proses pelabelan manual dilakukan untuk menilai sentimen dari masing-masing kelas, baik positif maupun negatif. Tujuan dari pelabelan adalah untuk memberikan nilai yang jelas dan konsisten pada sentimen yang diungkapkan dalam data. Setelah pelabelan selesai, akurasi dari penilaian sentimen tersebut akan dihitung dan dianalisis. Dengan kata lain, pelabelan manual ini merupakan langkah penting dalam memastikan bahwa sentimen yang terkandung dalam data dapat diukur dengan tepat, sehingga hasil analisis sentimen nantinya dapat dipercaya dan memiliki validitas tinggi.

4.2.4 Visualisasi

Penulis melakukan tahapan visualisasi dengan tujuan menyajikan data atau respons masyrakat ke dalam bentuk grafis. Adapun grafis yang dibuat oleh penulis berupa *Wordcloud. Visualisasi Wordcloud* adalah salah satu teknik visualisasi yang digunakan untuk menampilkan frekuensi kata dalam sebuah teks atau kumpulan teks. Wordcloud, atau awan kata, menyajikan kata-kata dalam berbagai ukuran dan warna sesuai dengan frekuensi atau pentingnya kata-kata tersebut dalam teks. Kata-kata yang muncul lebih sering akan ditampilkan dengan ukuran lebih besar dan mungkin warna yang lebih mencolok, sementara kata-kata yang muncul lebih jarang akan ditampilkan dengan ukuran lebih kecil. Penulis sudah membuat 2 *Visualisasi Wordcloud,* yaitu *Visualisasi Wordcloud* untuk sentimen positif, dan *Visualisasi Wordcloud* untuk sentimen negatif.

4.2.4.1 Wordcloud sentimen positif

Visualisasi Wordcloud sentimen positif dapat dilihat pada Gambar 4.10 berikut:

Gambar 4.10 *Wordcloud* sentimen positif

Dari Gambar *Wordcloud* diatas menampilkan 20 kata yang paling sering muncul dalam data sentimen positif. Kata-kata diatur menurut frekuensi kemunculannya, dengan ukuran yang lebih besar untuk kata-kata yang lebih sering muncul. Melalui *Wordcloud*, istilah yang paling umum

dalam konteks sentimen positif dapat diidentifikasi, memberikan wawasan tentang aspek-aspek yang paling dihargai atau disukai oleh pengguna.

4.2.4.2 *Wordcloud* sentimen negatif

Visualisasi Wordcloud untuk sentimen negatif dapat dilihat pada Gambar 4.11 berikut:

Gambar 4.11 *Wordcloud* sentimen negatif

Gambar *Wordcloud* tersebut menggambarkan 20 kata yang paling sering muncul dalam data sentimen negatif. Ukuran kata mencerminkan frekuensi kemunculannya, dengan kata-kata yang lebih sering muncul ditampilkan dalam ukuran yang lebih besar. *Wordcloud* memudahkan identifikasi istilah yang sering diungkapkan dalam konteks sentimen negatif, serta memberikan informasi tentang masalah atau keluhan yang umum dihadapi pengguna dalam data tersebut.

4.2.5 TF-IDF

Pada tahap TF-IDF, data teks yang digunakan berasal dari kolom clean_text setelah melalui proses preprocessing. Data teks dalam kolom *clean_text* awalnya diubah menjadi list dengan proses *word_tokenization*, yang bertujuan untuk memisahkan setiap kata menjadi token. Hal ini dilakukan untuk memberikan nilai bobot yang tepat pada setiap kata dalam analisis *TF-IDF*.

4.2.5.1 *Tokenize*

Melakukan proses tokenisasi kembali untuk dataset, menggunakan *`word_tokenize`* dari pustaka NLTK, merupakan langkah penting dalam pembobotan kata pada kolom *`clean_text`*. Proses tokenisasi dapat dilihat dalam Gambar 4.12.

	clean_text	sentimen	tokenized_text
0	ikut timnas beli jersey gapantes bilang fomo	positif	["ikut", "timnas", "beli", "jersey", "gapantes...
1	gue jersey suka timnas beli orisinal suka saya...	positif	["gue", "jersey", "suka", "timnas", "beli", "o...
2	putih eyy kesel perkara jersey pnyny timnas jelek	negatif	["putih", "eyy", "kesel", "perkara", "jersey",...
3	jersey timnas indonesia laris jersey timnas in...	positif	["jersey", "timnas", "indonesia", "laris", "je...
4	laris jersey timnas indonesia	positif	["laris", "jersey", "timnas", "indonesia"]

Gambar 4. 12 *Tokenize TF-IDF*

Gambar proses tokenisasi diatas melibatkan pemecahan teks menjadi unit-unit kecil yang disebut token, biasanya berupa kata-kata individu. Selama tokenisasi, teks mentah diuraikan berdasarkan spasi, tanda baca, atau aturan linguistik lainnya untuk menghasilkan daftar token. Hasil dari data yang sudah di tokenisasi dimasukkan ke kolom *tokenized_text.*

4.2.5.2 Pembobotan *Term Frequency* (TF)

Proses perhitungan TF menggunakan rumus persamaan (1) dari setiap kata dalam dataset di kolom *tokenized_clean_text.* Perhitungan TF memberikan gambaran tentang pentingnya suatu istilah dalam konteks dokumen tertentu. Hasilnya tersimpan dalam kolom *'TF''*. Gambar 4.13 merupakan hasil pembobotan TF.

	term	TF
0	0	{'"ikut"': 0.14285714285714285, '"timnas"': 0....
1	1	{'"gue"': 0.09523809523809523, '"jersey"': 0.0...
2	2	{'"putih"': 0.125, '"eyy"': 0.125, '"kesel"': ...
3	3	{'"jersey"': 0.2857142857142857, '"timnas"': 0...
4	4	{'"laris"': 0.25, '"jersey"': 0.25, '"timnas"'...
...	...	...
2397	2397	{'"erspo"': 0.16666666666666666, '"erigo"': 0....
2398	2398	{'"jersey"': 0.125, '"timnas"': 0.125, '"abad"...
2399	2399	{'"jersey"': 0.2, '"keluar"': 0.2, '"kelme"': ...
2400	2400	{'"hokky"': 0.125, '"timbang"': 0.125, '"bal"'...
2401	2401	{'"produsen"': 0.058823529411764705, '"pakai"'...

2402 rows × 2 columns

Gambar 4.13 *Term Frequency*

Penjelasan Gambar 4.13 yaitu TF dihitung dengan membagi jumlah kemunculan istilah tertentu dalam dokumen dengan jumlah total kata dalam dokumen tersebut. Tujuan dari pengukuran ini adalah untuk menentukan seberapa dominan atau signifikan istilah tersebut di dalam dokumen tertentu. Dengan menggunakan nilai TF, dapat mengetahui seberapa sering suatu istilah muncul, yang memberikan indikasi tentang pentingnya istilah tersebut dalam konteks dokumen tersebut.

4.2.5.3 Menghitung Document Frequency (DF)

Proses perhitungan DF menggunakan rumus persamaan (2), dengan cara menghitung DF *(Document Frequency)* dan menyajikannya kedalam dataframe/tabel agar mudah dibaca. Gambar 4.14 merupakan hasil perhitungan DF.

	term	DF
0	"ikut"	19
1	"timnas"	1217
2	"beli"	241
3	"jersey"	1865
4	"gapantes"	1
...	...	...
4999	"bangettttt"	1
5000	"bbayang"	1
5001	"majuu"	1
5002	"u"	1
5003	"indonesiaaa"	1

5004 rows × 2 columns

Gambar 4.14 *Document Frequency*

Gambar menyajikan tentang DF untuk mengukur berapa banyak dokumen dalam koleksi yang mengandung istilah tertentu. Dengan menghitung DF, dapat diketahui seberapa sering istilah tersebut muncul di berbagai dokumen, yang menunjukkan tingkat relevansi istilah tersebut. Semakin sering istilah muncul di dokumen-dokumen, semakin tinggi nilai DF yang diperoleh.

4.2.5.4 Menghitung TF-IDF

Menggunakan rumus persamaan (3) dan (4), menghitung bobot Term Frequency-Inverse Document Frequency (TF-IDF) untuk setiap kata dalam dataset dikolom Tweet. Gambar 4.15 adalah hasil perhitungan TF-IDF.

	term	TF	TF-IDF
0	0	{'"ikut"': 0.14285714285714285, '"timnas"': 0....	{'"ikut"': 0.6840463898694197, '"timnas"': 0.0...
1	1	{'"gue"': 0.09523809523809523, '"jersey"': 0.0...	{'"gue"': 0.3216512147976834, '"jersey"': 0.01...
2	2	{'"putih"': 0.125, '"eyy"': 0.125, '"kesel"': ...	{'"putih"': 0.46331494484177616, '"eyy"': 0.88...
3	3	{'"jersey"': 0.2857142857142857, '"timnas"': 0...	{'"jersey"': 0.07214417749503993, '"timnas"': ...
4	4	{'"laris"': 0.25, '"jersey"': 0.25, '"timnas"'...	{'"laris"': 1.4980743833529684, '"jersey"': 0....
...	...	...	...
2397	2397	{'"erspo"': 0.16666666666666666, '"erigo"': 0....	{'"erspo"': 0.09541671155435458, '"erigo"': 0....
2398	2398	{'"jersey"': 0.125, '"timnas"': 0.125, '"abad"...	{'"jersey"': 0.03156307765407997, '"timnas"': ...
2399	2399	{'"jersey"': 0.2, '"keluar"': 0.2, '"kelme"': ...	{'"jersey"': 0.05050092424652796, '"keluar"': ...
2400	2400	{'"hokky"': 0.125, '"timbang"': 0.125, '"bal"'...	{'"hokky"': 0.8356805892464774, '"timbang"': 0...
2401	2401	{'"produsen"': 0.058823529411764705, '"pakai"'...	{'"produsen"': 0.32243952409681664, '"pakai"':...

2402 rows × 3 columns

Gambar 4.15 Perhitungan *TF-IDF*

Gambar diatas menyajikan proses perhitungan nilai *TF* dengan *IDF*, Hasil dari TF dan IDF kemudian dikombinasikan untuk menghitung nilai *TF-IDF,* yang mencerminkan seberapa penting sebuah istilah dalam dokumen tertentu dibandingkan dengan seluruh koleksi dokumen. Nilai *TF-IDF* yang tinggi menunjukkan bahwa istilah tersebut sering muncul dalam dokumen tersebut namun jarang ditemukan di dokumen lain, menandakan kepentingan atau relevansi istilah itu dalam dokumen yang dimaksud. Metode tersebut memungkinkan penilaian yang lebih tepat terhadap relevansi istilah, sehingga dapat meningkatkan efektivitas dalam pencarian informasi dan analisis teks.

4.2.6 Klasifikasi Naïve Bayes

4.2.6.1 Data *Training*

Tahapan proses training data adalah mengatur penentuan sumbu X dan sumbu Y, pada penelitian ini penulis telah menentukan sumbu X adalah data dari kolom “clean_text”, sementara sumbu Y adalah data dari kolom “sentimen”. Proses pembagian disajikan pada Gambar 4.16.

```
#Set sumbu nilai X dan Y
X = df_nb1['clean_text']
y = df_nb1['sentimen']
```

Gambar 4.16 Penentuan Sumbu X dan Y

Setelah berhasil melakukan pembagian sumbu X dan Y dilanjutkan membagi data set, pada Gambar dibawah penulis telah membagi data dengan perbandingan 20% data test dan 80% data training atau sebanyak 1921 data training.

```
# Membagi dataset menjadi data latih dan data uji
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Gambar 4.17 Membagi dataset

Sebelum melakukan klasifikasi, data teks diekstraksi menggunakan *TfidfVectorizer* untuk menghitung bobot dari setiap kata dalam dokumen. Metode ini sangat penting karena membantu menentukan pentingnya sebuah kata dalam suatu dokumen dibandingkan dengan dokumen lainnya. Setelah data diekstraksi dan diubah menjadi representasi berbasis bobot, langkah berikutnya adalah mengklasifikasikan data tersebut. Algoritma *Naive Bayes*, khususnya MultinomialNB dari sklearn, digunakan untuk klasifikasi karena cocok untuk menangani data teks. Parameter alpha diatur untuk meningkatkan akurasi dengan menyempurnakan estimasi probabilitas, sedangkan opsi class balanced digunakan untuk memastikan distribusi data yang seimbang untuk setiap kategori sentimen, yang penting untuk menghindari bias dalam model. Gambar 4.18 adalah proses klasifikasi pada data *training*.

```
#KLASIFIKASI NAIVE BAYES
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report
# Membangun model tf-idf
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
# buat data yang akan diklasifikasi naive bayes menjadi balance
from sklearn.utils import class_weight
sample = class_weight.compute_sample_weight('balanced', y_train)
# Melatih model Naive Bayes
nb_classifier = MultinomialNB(alpha=0.5)
nb_classifier.fit(X_train_tfidf, y_train, sample_weight=sample)
# Memprediksi sentimen pada data uji
y_pred_train_nb = nb_classifier.predict(X_train_tfidf)
# Melihat classification report
print("Classification Report untuk Data Train:")
print(classification_report(y_train, y_pred_train_nb))
```

```
Classification Report untuk Data Train:
              precision    recall  f1-score   support

     negatif       0.97      0.95      0.96      1024
     positif       0.94      0.97      0.95       897

    accuracy                           0.96      1921
   macro avg       0.96      0.96      0.96      1921
weighted avg       0.96      0.96      0.96      1921
```

Gambar 4.18 Klasifikasi *training* data

Berdasarkan hasil klasifikasi data training dengan perbandingan 20% data test dan 80% data training, mendapatkan hasil accuracy 96%, nilai precision 97%, recall 95%, serta f1-score 96%.

4.2.6.2 Data *Testing*

Proses klasifikasi data testing yang disajikan pada Gambar 4.19 dibawah menggunakan perbandingan 20% data testing dan 80% data training atau menggunakan 481 data.

```
#KLASIFIKASI NAIVE BAYES
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report
# Membangun model tf-idf
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
# buat data yang akan diklasifikasi naive bayes menjadi balance
from sklearn.utils import class_weight
sample = class_weight.compute_sample_weight('balanced', y_train)
# Melatih model Naive Bayes
nb_classifier = MultinomialNB(alpha=0.5)
nb_classifier.fit(X_train_tfidf, y_train, sample_weight=sample)
# Memprediksi sentimen pada data uji
y_pred_nb = nb_classifier.predict(X_test_tfidf)
# Melihat classification report
print("Classification Report untuk Data Test:")
print(classification_report(y_test, y_pred_nb))
```

```
Classification Report untuk Data Test:
              precision    recall  f1-score   support

     negatif       0.91      0.86      0.88       237
     positif       0.87      0.91      0.89       244

    accuracy                           0.89       481
   macro avg       0.89      0.89      0.89       481
weighted avg       0.89      0.89      0.89       481
```

Gambar 4.19 Klasifikasi *testing* data

Hasil klasifikasi data testing dengan perbandingan 20% data test dan 80% data training, dari total 481 data yang telah diberi label dengan detail 237 data dengan label negatif dan 244 data dengan label positif, mendapatkan hasil mendapatkan hasil accuracy 89%, nilai precision 91%, recall 86%, serta f1-score 88%.

	Text	Prediksi	Aktual	Hasil	Confusion Matrix
685	ribut desain desainer beres ganti brandnya ken...	negatif	negatif	Benar	True Negative (TN)
111	ernanda tolol	negatif	negatif	Benar	True Negative (TN)
1512	nama baja tindak lepas desain dll erspo mills ...	positif	positif	Benar	True Positive (TP)
1651	ambil sound wave supporter filosofi desain des...	negatif	negatif	Benar	True Negative (TN)
741	jersey erspo desain bapuk enggak bakal buka de...	negatif	negatif	Benar	True Negative (TN)
...	...	...	...	...	...
43	yaiyalah bodoh main mantan timnas pakai jersey	negatif	negatif	Benar	True Negative (TN)
2080	berani hasil gede menang after all these kelas...	negatif	negatif	Benar	True Negative (TN)
610	bangkai kesini suka design jersey bikin erspo ...	negatif	negatif	Benar	True Negative (TN)
2320	cek jersey timnas indonesia grade orisinal hom...	positif	positif	Benar	True Positive (TP)
203	sumpah gue kemarin pengin beli jersey sepak bo...	positif	positif	Benar	True Positive (TP)

481 rows × 5 columns

Gambar 4.20 Hasil prediksi *testing* data

Gambar 4.20 diatas menyajikan tabel prediksi dari hasil klasifikasi data testing dengan perbandingan 20% data test dan 80% data training yang menampilkan perbandingan.

4.2.7 Evaluasi

4.2.7.1 Evaluasi Klasifikasi

Confusion matrix adalah alat evaluasi yang digunakan untuk menilai performa model klasifikasi. Tabel ini memperlihatkan perbandingan antara hasil prediksi model dengan label asli data. *Confusion matrix* menyediakan informasi mengenai jumlah prediksi yang benar dan salah di berbagai kategori. Hasil perhitungan *confusion matrix* pada testing data bisa dilihat pada Tabel 4.3.

Tabel 4.3 *Confusion Matrix testing* data

Kelas Aktual	Kelas Prediksi	
	Positif	Negatif
Positif	1067	194
Negatif	215	926

Hasil confusion matrix yang ditunjukkan pada tabel adalah TP = 1067, TN = 926, FP = 215, dan FN = 194. Setelah memperoleh nilai dari confusion matrix, langkah selanjutnya adalah melakukan perhitungan cross-validation. Cross-validation adalah metode evaluasi penting dalam klasifikasi yang membagi dataset menjadi beberapa subset atau fold, memungkinkan model dilatih dan diuji secara bergantian. Penulis telah menerapkan cross-validation pada data testing dengan melakukan perulangan sebanyak 10 kali untuk mendapatkan angka yang representatif. Setiap fold menghasilkan nilai yang berbeda, dan hasil perhitungan untuk 10 fold dapat dilihat pada Tabel 4.4 berikut.

Tabel 4.4 Hasil *Cross Validation*

Fold	**Accuracy**
Fold 1	80%
Fold 2	86%
Fold 3	82%
Fold 4	80%
Fold 5	80%
Fold 6	88%
Fold 7	84%
Fold 8	87%
Fold 9	78%
Fold 10	84%

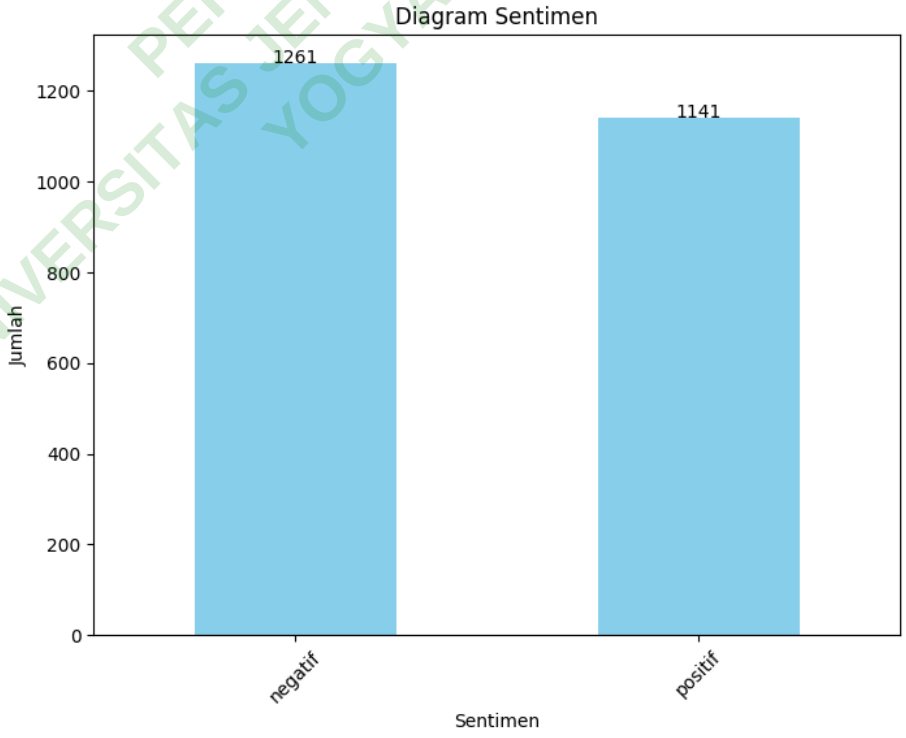
Cross-validation yang dilakukan sebanyak 10 kali menghasilkan rata-rata akurasi yang cukup memuaskan, yaitu 83%. Rincian hasil klasifikasi dapat dilihat pada Tabel 4.5.

Tabel 4.5 *Classification report* data testing

Jenis	Precison	Recall	F1-Score	Support
Negatif	0.83	0.85	0.84	1261
Positif	0.83	0.81	0.82	1141
Accuracy			0.83	2402
Macro Avg.	0.83	0.83	0.83	2402
Weighted Avg.	0.83	0.83	0.83	2402

4.2.7.2 Evaluasi Analisis

Setelah melakukan labeling data dapat menunjukkan terdapat 1141 data dengan sentimen positif dan 1261 data dengan sentimen negatif. Detail visualisasi distribusi sentimen dapat dilihat pada Gambar 4.21.



Gambar 4.21 Diagram sentimen

Sentimen positif terhadap data tweet membahas beberapa aspek, termasuk peningkatan kualitas *jersey* Timnas yang diklaim lebih baik daripada sebelumnya, keyakinan bahwa *jersey* tersebut membawa keberuntungan karena peforma meningkat signifikan sejak penggunaannya oleh Timnas Indonesia baik di level senior maupun Timnas U-23, serta antusiasme yang tinggi dari netizen terhadap rencana pembaruan desain *jersey* yang direncanakan diluncurkan kembali pada bulan Oktober 2024. Contoh lima data teratas yang mencerminkan sentimen positif dapat dilihat dalam Tabel 4.6.

Tabel 4.6 Data dengan sentimen positif

No	Data Tweet
1.	ikut timnas beli *jersey* gapantes bilang fomo
2.	*jersey* timnas indonesia laris *jersey* timnas indonesia
3.	*unpopular opinion jersey* erspo salah faktor tres positif timnas timnasu pssi
4.	kosistensi design *jersey* rubah dadak material logo rubah rubah *quality control* mumpuni pengin masuk dengar bagus evaluasi produk pakem bagus
5.	tolongghhgg erspo gue demen *jersey* lo pink plss juall gue beli

Sementara itu, data dengan label negatif membahas tentang berbagai kritik terhadap *jersey*, termasuk desain yang dianggap jelek, harga yang dianggap terlalu mahal, dan kontroversi seputar cuitan dari desainer Erspo yang memicu kemarahan netizen serta menarik kritik berkelanjutan. Kelima data teratas yang mengekspresikan sentimen negatif terdokumentasikan dalam Tabel 4.7.

Tabel 4.7 Data dengan sentimen negatif

No	Data Tweet
1.	putih eyy kesel perkara *jersey* pnyny timnas jelek
2.	*jersey* erspo harga *over price* mahal

3.	salah asi value *jersey* timnas indonesia era ganti nameset timpa nama jahit kain nama biar seru bahan jelek
4.	guys rumah cuman sisa *jersey* erspo kaos partai pilih pakai boikoterspo boikotmakna
5.	boikotmakna boikoterspo desain jelek model murah kayak *jersey* timnas singapore fix kagak kreatif erspo desain katro kagak kritik lepas kerjasama *erspo* malu nama bangsa negara *erspo*

4.2.7.3 Hasil Sentimen

Berdasarkan hasil analisis, dapat disimpulkan bahwa sentimen masyarakat terhadap jersey terbaru Timnas Indonesia produksi *Erspo* di media sosial *X* adalah negatif, dengan tingkat akurasi penelitian sebesar 83%. Penyebab sentimen negatif meliputi harga jersey yang tinggi, desain yang kurang menarik, dan perilaku buruk desainer *jersey*. Di sisi lain, sentimen positif muncul karena kualitas bahan jersey yang baik serta anggapan bahwa *jersey* terbaru membawa keberuntungan, mengingat performa Timnas Indonesia meningkat sejak menggunakan *jersey* tersebut.