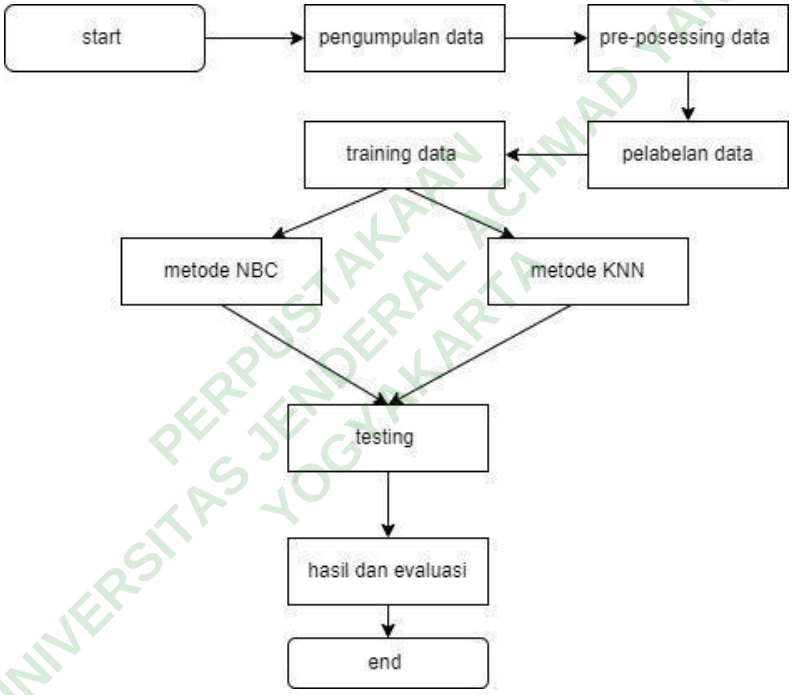
BAB 3
METODE PENELITIAN

Penelitian ini adalah penelitian kuantitatif, metode penelitian analisis sentimen tentang pengguna jasa ekspedisi yang diambil dari opini masyarakat pada *platform* Twitter yang menghasilkan sentimen positif dan sentimen negatif. Adapun janannya penelitian yang dilakukan dalam penelitian ini dapat dilihat pada Gambar 3.1



Gambar 3.1 Jalannya Penelitian

Tahapan awal dari jalannya penelitian ini adalah pengumpulan data didasari masalah yang ada di masyarakat dari *platform* Twitter, kemudian akan dilanjutkan per-*processing* data untuk menyeleksi data dari singkatan, stiker dan bahasa daerah, kemudian data akan dikelompokkan berdasarkan data yang diinginkan, setelah data dikelompokkan kemudian akan dilakukan pelabelan yang dilakukan secara manual dan akan dilakukan *training* menggunakan 2 metode yaitu NBC dan KNN dan tahapan terakhir adalah *testing* untuk menghasilkan data yang menggambarkan sentimen jasa ekspedisi di *platform* media sosial Twitter.

Berikut ini adalah bahan, alat, dan jalannya penelitian analisis sentimen pengguna jasa ekspedisi, tahapan penelitian untuk menyelesaikan proses analisis sentimen menggunakan data opini masyarakat yang ada di Twitter.

3.1 BAHAN DAN ALAT PENELITIAN

Bahan yang akan dibutuhkan dalam penelitian ini adalah data *tweet, re-tweet* dan komentar di Twitter yang berkaitan dengan layanan jasa ekspedisi. Penelitian ini mengambil data dari bulan April sampai Mei tahun 2022 dengan kata kunci JNT, JNE dan Pos Indonesia.

Alat yang digunakan dalam penelitian ini adalah komputer dengan spesifikasi cukup untuk menjalankan sistem operasi dan perangkat lunak pengembangan serta koneksitas *Internet*.

Sistem Operasi dan program-program aplikasi yang dipergunakan dalam dalam pengembangan aplikasi ini adalah:

1. Sistem Operasi: Windows 8 atau lebih baru.
2. Bahasa pemrograman python 3.10.4
3. Microsoft Excel 2016
4. Google Collab
5. Sublime Text

3.2 JALAN PENELITIAN

Pada penelitian ini menggunakan softwere Anaconda 3 dan Google Collab dengan bahasa pemrograman Python dan *framework* Flask untuk pengambilan data dari Twitter yang berkaitan dengan jasa ekspedisi. Adapun tahapan pada jalannya penelitian ini yaitu:

3.2.1 Tahap Pengumpulan Data

Pengambilan data dijalankan di website google collab menggunakan bahasa pemrograman python dengan library snscrape, *liblary* ini tidak membutuhkan akses token untuk pengambilan datanya, data yang diambil bisa lebih dari 7 hari. Data yang diambil kemudian disimpan dan ditampilkan dalam bentuk ms.excel. Data yang diambil adalah data *tweet* dan *retweet* dengan kata

kunci “JNT, JNE dan Pos Indonesia” data yang diperoleh JNT 5.000, JNE 5.000 dan Pos Indonesia 3.812.

Hasil data yang telah diperoleh disimpan dengan format CSV untuk mempermudah dalam pre-*pocessing*. Pengumpulan data ini dilakukan dari tanggal 28 April 2022 sampai tanggal 24 mei 2022, dengan kategori *DateTime*, *Tweet Id*, *Text* dan *Username*, kode yang digunakan dalam pengambilan data dapat dilihat pada kode:

```
import snscrape.modules.twitter as sntwitter
import pandas as pd

maxTweets = 5000

tweets_list = []

for i,tweet in enumerate(sntwitter.TwitterSearchScraper('JNE since:20
22-04-28 until:2022-05-24 lang:id').get_items()):
    if i>maxTweets:
        break
    tweets_list.append([tweet.date, tweet.username, tweet.content])

tweets_df = pd.DataFrame(tweets_list, columns=['Datetime', 'Username'
, 'Text'])
tweets_df = tweets_df.drop_duplicates()
```


Pemanggilan *tweets list* juga berfungsi untuk menyimpan file yang didapat yang ditampilkan dalam bentuk *ms.excel*. Dapat dilihat hasil pengambilan data pada Tabel 3. 1

Tabel 3.1 Hasil Pengambilan Data

No	Waktu	Username	Teks
1	24/05/2022	they call me Tiago	b'heh ini knp jne jadi cepet bener sekarang, mantep \xf0\x9f\x98\xad https://t.co/20xtoXCP7X'
2	24/05/2022	Stalking.admirer	b'Pada tgl 07 mei 2022 hari sabtu sore sekitar jam 16.00-18.00 aku suruh adek aku ke JNE cimone buat kirim paket ke t\xe2\x80\xa6 https://t.co/9l6F9mOi3B'

3	24/05/2022	nana	b'Jne ini ngapa dah ngestuck, paket udah mau seminggu di jakarta. Jadwal ini maksudnya apa ya? Kalo ilang paketnya gm\xe2\x80\xa6 https://t.co/Fw0roekWTA'
4	24/05/2022	wid	b'selalu kesel deh kalo pake jne'
5	24/05/2022	Dhea F	b'Demi allah najis bgt jne tuh, telfon cs nya kaya gak guna sama sekali jawabannya ngalor ngidul. Paketan gue ada di\xe2\x80\xa6 https://t.co/SBojnkudds'
6	24/05/2022	jasmineðŸŒ»	b'ini dia lg sibuk cari jne yg ada packing kayunya dan gue nih ntar harus jemput dia mana gue susaahh kalo nyetir pas ujan woy'
7	24/05/2022	Unpadfess (Discord di Bio)	b'- halo, ada yg tau ga jne di hegar tutup jamber?? makasihh'
8	24/05/2022	de.	b'@paket_jne lahh..datengnya sore?'
9	24/05/2022	ca.	b'gila njing jne lama bgt kek siput'
10	24/05/2022	JNE DEPOK	b'kami melayani kiriman anda sampai ke penjuru nusantara dan pelosok desa... percayakan pada JNE Juanda Depok saja yukk.'

Dari Tabel 3.1 menunjukan bahwa hasil pengambilan data masih banyak bagian yang tidak diperlukan, karena tidak diperlukan dalam tahap analisis sentimen maka bagian-bagian tersebut harus dihapus. Data yang telah diambil akan melalui tahap pre-pocessing agar mendapatkan hasil yang sesuai.

3.2.2 Tahapan Prepocessing

Tahapan ini akan dilakukan pemrosesan pembersihan data dari noise dan mengubah bentuk ke bentuk dasa (kata dasar). Tujuannya untuk perbaikan data dan menormalisasikan data, maka pada tahapan ini akan dilakukan proses *creansing* dan *formalization* dengan harapan meningkatkan akurasi.

Tahapan *Preprocessing* dalam tahapan ini dijalankan di google collab menggunakan bahasa pemrograman python. Tahapan ini dilakukan untuk memperbaiki dan menormalisasikan data dari singkatan, url, emoji, bahasa daerah, menghapus kata yang tidak sopan atau kata kasar.

1. Removal atau Cleaning Data

Tahapan ini membersihkan data dari bagian-bagian yang tidak diperlukan. Tujuannya untuk menghidari dublikat data, kode yang digunaakan dalam proses *removal* dapat dilihat sebagai berikut:

```
def url_remove(Teks):
    t = re.sub(r'http\S+', '',Teks)
    return t

def url (Teks):
    t = re.sub('((www\.[^\s]+)|(https?://[^\s]+))','URL', Tek
s)
    return t

def punc_remove(Teks):
    t = re.sub(r'[^\w\s]','', Teks)
    return t

def rt_remove(Teks):
    t = re.sub(r'RT[\s]+', '', Teks)
    return t

def number_remove(Teks):
    t = re.sub('[0-9]+', '', Teks)
    return t

def slang_remove(Teks):
    t = re.sub(r'\\n', " ", Teks)
    return t

def regex_remove(Teks):
    t = re.sub("b'", " ", Teks)
    return t

def remove_user(Teks):
    t = re.sub('@[^\s]+','', Teks)
    return t

def hashtag_remove(Teks):
    reg = "#(\w+:\/\/S+)"
    return re.sub(reg, " ", Teks)
```

Kode diatas digunakan untuk menghapus karakter yang tidak berpengaruh dalam proses klasifikasi seperti *hasthteg*(#), *mention*(@),

url(http:\\) dan karakter huruf lainnya. Setelah dilakukan prose cleaning data maka data akan bersih dari noise, dublikat data dan karakter huruf.

2. Tokenizing

Tokenizing adalah pemecahan kalimat menjadi kata, menghapus simbol dan menghilangkan tanda baca dalam kalimat, sehingga mempermudah dalam proses analisis. Kode yang digunakan dalam tokenizing dapat dilihat sebagai berikut:

```
cleaned = []
def clean_text(Teks):
    for i in Teks:
        cleaned.append(url_remove(punc_remove(number_remove(r
 emove_user(regex_remove(hashtag_remove(rt_remove(slang_remov
 e(
            re.sub("[\n\r\t\xa0]"," ",i).strip()))))))))))
```

Fungsi kode diatas digunakan dalam tahapan *tokenizing*. Hasil dari tahapan *cleaning* dan *tokenizing* dapat dilihat pada Gambar 3.2

index	Teks	teks
0	b'heh ini knp jne jadi cepet bener sekarang, mantep \xf0\x9f\x98\xad https://t.co/20xtoXCP7X'	heh ini knp jne jadi cepet bener sekarang mantep xfxfxxad
1	b'Pada tgl 07 mei 2022 hari sabtu sore sekitar jam 16.00-18.00 aku suruh adek aku ke JNE cimone buat kirim paket ke t\xe2\x80\xa6 https://t.co/9l6F9mOi3B'	Pada tgl mei hari sabtu sore sekitar jam aku suruh adek aku ke JNE cimone buat kirim paket ke txexxa
2	b'Jne ini ngapa dah ngestuck, paket udah mau seminggu di jakarta. Jadwal ini maksudnya apa ya? Kalo ilang paketnya gm\xe2\x80\xa6 https://t.co/Fw0roekWTA'	Jne ini ngapa dah ngestuck paket udah mau seminggu di jakarta Jadwal ini maksudnya apa ya Kalo ilang paketnya gmxexxa
3	b'selalu kesel deh kalo pake jne'	selalu kesel deh kalo pake jne
4	b'Demi allah najis bgt jne tuh, telfon cs nya kaya gak guna sama sekali jawabannya ngalor ngidul. Paketan gue ada di\xe2\x80\xa6 https://t.co/SBojnkudds'	Demi allah najis bgt jne tuh telfon cs nya kaya gak guna sama sekali jawabannya ngalor ngidul Paketan gue ada dixexxa
5	b'ini dia lg sibuk cari jne yg ada packing kayunya dan gue nih ntar harus jemput dia mana gue susaahh kalo nyetir pas ujan woy'	ini dia lg sibuk cari jne yg ada packing kayunya dan gue nih ntar harus jemput dia mana gue susaahh kalo nyetir pas ujan woy
6	b'- halo, ada yg tau ga jne di hegar tutup jamber?? makasihh'	halo ada yg tau ga jne di hegar tutup jamber makasihh
7	b'@paket_jne lahh..datengnya sore?'	lahhdatengnya sore
8	b'gila njing jne lama bgt kek siput'	gila njing jne lama bgt kek siput
9	b'kami melayani kiriman anda sampai ke penjuru nusantara dan pelosok desa... percayakan pada JNE Juanda Depok saja yukk.'	kami melayani kiriman anda sampai ke penjuru nusantara dan pelosok desa percayakan pada JNE Juanda Depok saja yukk

Gambar 3.2 Hasil *Tokenizing*

3. Case Folding

Tahapan *case folding* adalah perubahan bentuk karakter huruf dari bentuk awal menjadi huruf kecil, kode yang digunakan dalam *lowercase* dapat dilhat sebagai berikut:

```
def lowercase():
    lower_word = df['Teks'].str.lower()
    return lower_word

lower_tweet = lowercase()
```

Fungsi dari *lowercase* yaitu untuk menyamaratakan penggunaan huruf, karena banyak penulisan yang tidak konsisten dalam penulisan opini dan Sebagian besar penulisan menggunakan huruf kecil. Berikut ini hasil yang diperoleh dari *case floding* ditampilkan pada Gambar 3.3

```
0    b'heh ini knp jne jadi cepet bener sekarang, m...
1    b'pada tgl 07 mei 2022 hari sabtu sore sekitar...
2    b'jne ini ngapa dah ngestuck, paket udah mau s...
3                    b'selalu kesel deh kalo pake jne'
4    b'demi allah najis bgt jne tuh, telfon cs nya ...
5    b'ini dia lg sibuk cari jne yg ada packing kay...
6    b'- halo, ada yg tau ga jne di hegar tutup jam...
7                  b'@paket_jne lahh..datengnya sore?'
8               b'gila njing jne lama bgt kek siput'
9    b'kami melayani kiriman anda sampai ke penjuru...
Name: Teks, dtype: object
```

Gambar 3.3 Hasil *Case Folding*

4. Stopword Removal

Tahapan *stopword removal* ini bertujuan untuk menyaring kata asing atau menghilangkan kata dasar sebelum data diklasifikasi, kemudian menghapus teks yang tidak berhubungan dalam analisis. Kode yang digunakan dalam *stopword removal* dapat dilihat sebagai berikut:

```
from Sastrawi.StopWordRemover.StopWordRemoverFactory import S
topWordRemoverFactory
```

```
factory = StopWordRemoverFactory()
 more_stopword = ['rt', 'anjir', 'anjing', 'monyet', 'jancok
    ', 'tolol', 'bego', 'goblok', 'njing']
 stopword = factory.create_stop_word_remover()
 stopwords = factory.get_stop_words()+more_stopword
 print(stopwords)
```

Stopword remove menghasilkan kata-kata yang tidak diperlukan dan akan dihapus disesuaikan dengan *library* sastrawi. Daftar kata yang dihasilkan di tahap *stopword* ditapilkan pada Tabel 3.2

Tabel 3.2 *Stopword Removal*

```
['yang', 'untuk', 'pada', 'ke', 'para', 'namun',
'menurut', 'antara', 'dia', 'dua', 'ia', 'seperti',
'jika', 'jika', 'sehingga', 'kembali', 'dan',
'tidak', 'ini', 'karena', 'kepada', 'oleh', 'saat',
'harus', 'sementara', 'setelah', 'belum', 'kami',
'sekitar', 'bagi', 'serta', 'di', 'dari', 'telah',
'sebagai', 'masih', 'hal', 'ketika', 'adalah',
'itu', 'dalam', 'bisa', 'bahwa', 'atau', 'hanya',
'kita', 'dengan', 'akan', 'juga', 'ada', 'mereka',
'sudah', 'saya', 'terhadap', 'secara', 'agar',
'lain', 'anda', 'begitu', 'mengapa', 'kenapa',
'yaitu', 'yakni', 'daripada', 'itulah', 'lagi',
'maka', 'tentang', 'demi', 'dimana', 'kemana',
'pula', 'sambil', 'sebelum', 'sesudah', 'supaya',
'guna', 'kah', 'pun', 'sampai', 'sedangkan',
'selagi', 'sementara', 'tetapi', 'apakah',
'kecuali', 'sebab', 'selain', 'seolah', 'seraya',
'seterusnya', 'tanpa', 'agak', 'boleh', 'dapat',
'dsb', 'dst', 'dll', 'dahulu', 'dulunya', 'anu',
'demikian', 'tapi', 'ingin', 'juga', 'nggak',
'mari', 'nanti', 'melainkan', 'oh', 'ok',
'seharusnya', 'sebetulnya', 'setiap', 'setidaknya',
'sesuatu', 'pasti', 'saja', 'toh', 'ya', 'walau',
'tolong', 'tentu', 'amat', 'apalagi',
'bagaimanapun', 'rt', 'anjir', 'anjing', 'monyet',
'jancok', 'tolol', 'bego', 'goblok']
```

Setelah data berhasil ditampilkan, selanjutnya akan dilakukan penghapusan pada data yang ada dalam daftar *remove*. Kode yang digunakan dalam *remove stopwords* dan hasil dari proses *remove stopwords* dapat dilihat pada Gambar 3.4

```
def removeStopWords(Teks):
 clean_word_list = [word for word in Teks.split() if word not
in stopwords]
```

```
    return clean_word_list

stopwords_tweet = lower_tweet.apply(removeStopWords)

print(stopwords_tweet)
```

```
0    [b'heh, knp, jne, jadi, cepet, bener, sekarang...
1    [b'pada, tgl, 07, mei, 2022, hari, sabtu, sore...
2    [b'jne, ngapa, dah, ngestuck,, paket, udah, ma...
3             [b'selalu, kesel, deh, kalo, pake, jne']
4    [b'demi, allah, najis, bgt, jne, tuh,, telfon,...
5    [b'ini, lg, sibuk, cari, jne, yg, packing, kay...
6    [b'-, halo,, yg, tau, ga, jne, hegar, tutup, j...
7              [b'@paket_jne, lahh..datengnya, sore?']
8         [b'gila, njing, jne, lama, bgt, kek, siput']
9    [b'kami, melayani, kiriman, penjuru, nusantara...
Name: Teks, dtype: object
```

Gambar 3.4 Hasil *Stopword Removal*

5. Stemming

Tahapan *stemming* adalah proses untuk menemukan kata dasar dari sebuah kata. Kode yang digunakan dalam *stemming* dapat dilihat sebagai berikut:

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

def stemmed_wrapper(term):
    return stemmer.stem(term)

term_dict = {}

for document in stopwords_tweet:
    for term in document:
        if term not in term_dict:
            term_dict[term] = " "

print(len(term_dict))
print("------------------------")

for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
    print(term,":" ,term_dict[term])

print(term_dict)
print("------------------------")

    def get_stemmed_term(document):
        return [term_dict[term] for term in document]
```

```
stem_tweet = stopwords_tweet.apply(get_stemmed_term)

print(stem_tweet)
```

Stemming dilakukan untuk mengubah kata menjadi kata dasar sesuai dengan Bahasa Indonesia yang baik dan benar dengan menggunakan *library* stimmerFactory.

6. Normalization

Tahapan ini dilakukan untuk membenarkan kata yang tidak sesuai atau salah, memperbaiki dari singkatan dan bahasa daerah. Kode yang digunakan dalam *normalization* dapat dilihat sebagai berikut:

```
normalizad_word = pd.read_excel("normal jnt.xlsx")

normalizad_word_dict = {}

for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]

def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_wo
rd_dict else term for term in document]

normal_tweet = stem_tweet.apply(normalized_term).str.join(" ")

print(normal_tweet)
data_preprocess = normal_tweet

data_preprocess
tt={"Datetime":data['Datetime'],"Username":data['Username'] ,"T
ext":data_preprocess}

df=pd.DataFrame(tt)
```

Kata-kata yang sudah dibenarkan di *imporkan* kembali untuk membenarkan kata yang sebelumnya. Contoh data sebelum dan sesudah di normasilasikan dapat dilihat pada Tabel 3.3

Tabel 3.3 Normalisasi kata

Before	After
knp	Kenapa
ngapa	Mengapa
udah	Sudah
kesel	Kesal
bgt	Banget
gak	Tidak
ngalor	Utara
ngidul	Selatan
gue	Gwe
lg	Lagi
yg	Yang
ntar	Nanti
ga	Tidak
kek	Seperti

Setelah kata perbaiki sistem akan otomatis mengganti kata-kata yang salah atau tidak baku. Hasil normalisasi dapat dilihat pada Tabel 3.4

Tabel 3.4 Hasil Normalisasi

Teks
heh kenapa jne jadi cepet bener sekarang mantep xf0 x9f x98 xad https t co 20xtoxcp7x
pada tgl 07 mei 2022 hari sabtu sore jam 16 00-18 00 aku suruh adek aku jne cimone buat kirim paket t xe2 x80 xa6 https t co 9l6f9moi3b
jne mengapa dah ngestuck paket sudah mau minggu jakarta jadwal maksud apa ya kalo ilang paket gm xe2 x80 xa6 https t co fw0roekwta
selalu kesal deh kalo pake jne
demi allah najis banget jne tuh telfon cs nya kaya tidak sama sekali jawab utara selatan paket gwe di xe2 x80 xa6 https t co sbojnkudds
ini lagi sibuk cari jne yang packing kayu gwe nih nanti jemput mana gwe susaahh kalo nyetir pas ujan woy
halo yang tau ga jne hegar tutup jamber makasihh
paket jne lahh datengnya sore
gila jne lama banget seperti siput

kami layan kirim penjuru nusantara pelosok desa percaya jne juanda depok yukk
heh kenapa jne jadi cepet bener sekarang mantep xf0 x9f x98 xad https t co 20xtoxcp7x

3.2.3 Pelabelan Manual

Proses pelabelan dilakukan secara manual untuk mengklasifikasikan kata atau kalimat terhadap dokumen, sehingga dapat dianalisis lebih lanjut. Label yang digunakan menentukan hasil yang bersifat positif dan negatif.

Data yang sudah dinormalisasikan kemudian memasuki tahap *training*. Pelabelan yang diberikan pada data bersifat negatif atau positif, setiap ekspedisi akan diberikan label yang dilakukan di Ms.excel. jumlah data yang diberi label sebanyak 1.520. keterangan dalam pelabelan negatif dengan skor 0 dan positif skor 1. Hasil pelabelan manual dapat dilihat pada Gambar 3.5

	No	Skor	Kelas	Teks
0	1	1	Positif	aku pernah x ngirim barang pake jne yg pertam...
1	2	0	Negatif	Idk but kyknya jne gtgt jnt sih
2	3	1	Positif	Hai kak Aries mohon maaf untuk informasinya s...
3	4	1	Positif	Paling best sicepat sama JNE sih jalo di gw
4	5	1	Positif	kalo mau saya kirim jne aja yaa saya trauma ...
...	...	...	...	...
394	1767	1	Positif	Oke kak Lukman adm bantu cek DMnya ya Thank...
395	1772	1	Positif	Hai kak mohon maaf atas kejadian tsb ya kak ...
396	1778	1	Positif	Siang kak Informasi perihal paketnya akan ka...
397	1783	1	Positif	Samasama ya kak Bayu senang mendnegar kabar b...
398	1784	1	Positif	aku biasa ke jne jnt dikasihh tapi selalu min...

399 rows × 4 columns

Gambar 3.5 Hasil Pelabelan Manual

3.2.4 Training Data

Metode *training* data ini digunakan untuk melatih dan memprediksi data yang nantinya akan digunakan dalam pembelajaran

data uji. *Training* data atau data latih adalah model yang digunakan untuk melakukan klasifikasi. *Training* data terdiri dari 1.120 data yang dibagi menjadi 560 positif dan 560 negatif, *training* data dilakukan dengan menggunakan metode yaitu NBC.

1. TF-IDF

Trem frequency-inverse document frequency atau TF-IDF merupakan metode yang digunakan untuk menentukan nilai frekuensi dalam dokumen. Pada tahapan ini dilakukannya pembobotan disetiap kata dan dilakukannya pemecahan kata, berikut ini adalah proses yang dilakukan dalam TF-IDF. Dokumen yang digunakan dapat dilihat pada Tabel 3.5

Tabel 3.5 Dokumen TF-IDF

Dokumen (d)	Kalimat
d1	Pernah saya pak kirim pake jne aja terus tambahin asuransi
d2	biasa pengiriman JNE Trucking berapa lama ya dari Jakarta Pusat ke Kalimantan Selatan Via track record ga gerak selama hari
d3	Aku pernah pakai pos Indonesia
d4	supir JNT senang sekarang ada tol
d5	supir Pos Indonesia senang sekarang ada tol

Tabel 3. 5 merupakan contoh dokumen yang diambil untuk melakukan perhitungan tf-Idf yang dilakukan secara manual. Dengan menggunakan 5 dokumen yaitu d1, d2, d3, d4 dan d5. Perhitungan TF memerlukan beberapa komponen seperti t atau kata, d atau dokumen dan df atau banyak kata yang muncul. Perhitungan TF dapat dilihat pada Tabel 3.6

Tabel 3.6 Perhitungan TF

T	d1	d2	d3	d4	d5	Df
Pernah	1		1			2
Saya	1					1
Pak	1					1
Kirim	1					1
Pake	1		1			2
Jne	1	1				2
Aja	1					1
Terus	1					1
tambahin	1					1
asuransi	1					1
Biasa		1				1
pengiriman		1				1
tracking		1				1
Berapa		1				1
Lama		1				1
Ya		1				1
Dari		1				1
jakarta		1				1
Pusat		1				1
Ke		1				1
kalimantan		1				1
selatan		1				1
Via		1				1
Track		1				1
Record		1				1
Ga		1				1
gerak		1				1
selama		1				1
Hari		1				1
Aku			1			1
Pos			1		1	2
indonesia			1		1	2
Supir				1		1
Jnt				1		1

seneng				1	1	2
sekarang				1	1	2
Ada				1	1	2
Tol				1	1	2

Pada Tabel menjelaskan setiap kata yang ada didokumen. Pada perhitungan IDF memerlukan beberapa kompone seperti t atau kata dan df, idf adalah perhitungan kata dengan N atau banyaknya dokumen. Perhitungan idf dapat dilihat pada Tabel 3.7

Tabel 3.7 Perhitungan IDF

Kata(t)	Df	Idf	idf (N=5)	idf(N=1500)
Pernah	2	0,5	0,39794	2,87506126
Saya	1	1	0,69897	3,17609126
Pak	1	1	0,69897	3,17609126
Kirim	1	1	0,69897	3,17609126
Pake	2	0,5	0,39794	2,87506126
Jne	2	0,5	0,39794	2,87506126
Aja	1	1	0,69897	3,17609126
Terus	1	1	0,69897	3,17609126
Tambahin	1	1	0,69897	3,17609126
Asuransi	1	1	0,69897	3,17609126
Biasa	1	1	0,69897	3,17609126
Pengiriman	1	1	0,69897	3,17609126
Tracking	1	1	0,69897	3,17609126
Berapa	1	1	0,69897	3,17609126
Lama	1	1	0,69897	3,17609126
Ya	1	1	0,69897	3,17609126
Dari	1	1	0,69897	3,17609126
Jakarta	1	1	0,69897	3,17609126
Pusat	1	1	0,69897	3,17609126
Ke	1	1	0,69897	3,17609126
Kalimantan	1	1	0,69897	3,17609126
Selatan	1	1	0,69897	3,17609126
Via	1	1	0,69897	3,17609126
Track	1	1	0,69897	3,17609126
Record	1	1	0,69897	3,17609126

Ga	1	1	0,69897	3,17609126
gerak	1	1	0,69897	3,17609126
Selama	1	1	0,69897	3,17609126
Hari	1	1	0,69897	3,17609126
Aku	1	1	0,69897	3,17609126
Pos	2	0,5	0,39794	2,87506126
Indonesia	2	0,5	0,39794	2,87506126
Supir	1	1	0,69897	3,17609126
Jnt	1	1	0,69897	3,17609126
Seneng	2	0,5	0,39794	2,87506126
Sekarang	2	0,5	0,39794	2,87506126
Ada	2	0,5	0,39794	2,87506126
Tol	2	0,5	0,39794	2,87506126

Table menjelaskan perhitungan IDF secara manual menggunakan rumus idf=($\frac{N}{df}$). Perhitungan selanjutnya dapat dilihat pada Tabel 3.8

Tabel 3.8 Hasil perhitungan TF-IDF

Kata (t)	d1	d2	d3	d4	d5
Pernah	0,39794		0,39794		
Saya	0,69897				
Pak	0,69897				
Kirim	0,69897				
Pake	0,39794				
Jne	0,39794	0,39794			
Aja	0,69897				
Terus	0,69897				
tambahin	0,69897				
asuransi	0,69897				
Biasa		0,69897			
pengiriman		0,69897			
tracking		0,69897			
Berapa		0,69897			
Lama		0,69897			
Ya		0,69897			
Dari		0,69897			
jakarta		0,69897			

Pusat		0,69897			
Ke		0,69897			
kalimantan		0,69897			
selatan		0,69897			
Via		0,69897			
Track		0,69897			
Record		0,69897			
Ga		0,69897			
gerak		0,69897			
selama		0,69897			
Hari		0,69897			
Aku			0,69897		
Pos			0,39794		0,39794
indonesia			0,39794		0,39794
Supir				0,69897	
Jnt				0,69897	
seneng				0,39794	0,39794
sekarang				0,39794	0,39794
Ada				0,39794	0,39794
Tol				0,39794	0,39794

Pada Tabel 3. 8 adalah hasil dari perkalian dari TF dan IDF yang dilakukan secara manual.

Pada penelitian ini dilakukan penghitungan klasifikasi menggunakan fitur ekstraksi TF-IDF yang menghasilkan perhitungan kata secara otomatis pada kata atau dokumen training. Menggunakan *library* sklearn pada pemrograman python dengan model *tfidfvectorizer* untuk perhitungan secara otomatis. Kode yang digunakan dapat dilihat sebagai berikut:

```
from sklearn.feature_extraction.text import TfidfVectorizer

s1 = "jnt kebiasaan bgt gini udah ditungguin jugaaa "
```

```
s2 = "Pket saya udh  minggu belum sampai"

vect = TfidfVectorizer()
X = vect.fit_transform([s1, s2])

X.toarray()
```

Berikut ini adalah hasil dari TF-IDF yang dilakukan di mesin NBC yang dapat dilihat pada Gambar 3.6

```
array([[0.        , 0.        , 0.39204401, 0.        , 0.        ,
        0.39204401, 0.39204401, 0.39204401, 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.39204401, 0.27894255, 0.        , 0.        , 0.39204401,
        0.        ],
       [0.26255634, 0.26255634, 0.        , 0.26255634, 0.26255634,
        0.        , 0.        , 0.        , 0.26255634, 0.26255634,
        0.26255634, 0.26255634, 0.26255634, 0.26255634, 0.26255634,
        0.        , 0.186811  , 0.26255634, 0.26255634, 0.        ,
        0.26255634]])
```

Gambar 3.6 Hasil Tf-Idf secara otomatis

Fungsi ini diambil dari library sklearn dengan model TfidfVectorizer berfungsi ini untuk mempertimbangkan seluruhan data yang berfrekuensi dengan indeks kosa kata lain. Kemudian training ke dalam metode untuk mengetahui akurasi yang di dapat dari pemodelan, kode yang digunakan dapat dilihat sebagai berikut:

```
from sklearn.metrics import accuracy_score, f1_score, confus
ion_matrix

print("Accuracy: {:.2f}%".format(accuracy_score(y_test, y_pr
ed) * 100))
print("\nF1 Score: {:.2f}".format(f1_score(y_test, y_pred, a
verage='weighted') * 100))
print("\nCOnfusion Matrix:\n", confusion_matrix(y_test, y_pr
ed))
```

Fungsi ini diambil dari library sklearn dengan model accuracy score, f1 score dan confusion matrix yang masing-masing memiliki fungsi yang berbeda, namun saling ketergantungan. Accuracy score berfungsi untuk menghitung akurasi dari sampel data, f1 score berfungsi untuk menghitung rata-rata dari setiap kalimat multitasking

dan confusion matrix berfungsi mengukur kinerja algoritma dan membandingkan hasil yang diperoleh. Kemudian algoritma akan melakukan pembagian secara acak untuk mendapatkan akurasi model, kode yang digunakan dapat dilihat sebagai berikut:

```
from sklearn.model_selection import ShuffleSplit

X = df.Text
y = df.Skor

ss = ShuffleSplit(n_splits=10, test_size=0.2)
sm = SMOTE()

accs = []
f1s = []
cms = []
```

Perhitungan yang dilakukan secara acak menggunakan library sklaern dengan model ShufflenShit. Pemodelan ini dilakukan untuk menghasilkan akurasi model menggunakan perhitungan cross validation. Kode yang digunakan untuk membuat perhitungan cross validation dapat dilihat sebagai berikut:

```
for train_index, test_index in ss.split(X):

    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Fit vectorizer and transform X train, then transform X
test
    X_train_vect = vect.fit_transform(X_train)
    X_test_vect = vect.transform(X_test)

    # Oversample
    X_train_res, y_train_res = sm.fit_resample(X_train_vect,
y_train)

    # Fit Naive Bayes on the vectorized X with y train labels
,
    # then predict new y labels using X test
    nb.fit(X_train_res, y_train_res)
    y_pred = nb.predict(X_test_vect)

         # Determine test set accuracy and f1 score on this f
         old using the true y labels and predicted y labels
    accs.append(accuracy_score(y_test, y_pred))
    f1s.append(f1_score(y_test, y_pred, average='weighted'))
```

```
cms.append(confusion_matrix(y_test, y_pred))
```

Setelah melakukan perhitungan model, kemudian akan data akan ditampilkan berdasarkan akurasi yang diperoleh. Perintah yang digunakan untuk menapilkan data, dapat dilihat sebagai berikut:

```
print("\nAverage accuracy across folds: {:.2f}%".format(sum(
accs) / len(accs) * 100))
print("\nAverage F1 score across folds: {:.2f}%".format(sum(
f1s) / len(f1s) * 100))
print("\nAverage Confusion Matrix across folds: \n {}".forma
t(sum(cms) / len(cms)))
```

Perhitungan *cross validation* dilakukan perhitungan sebanyak 10 (sepuluh) kali percobaan, untuk mengetahui kinerja dari model algoritma yang digunakan. Selanjutnya akan dilakukan pembuatan model *pipeline* dengan *library* yang digunakan dapat dilihat sebagai berikut:

```
import os
import pickle
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer
```

Fungsi dari *library* sklearn menggunakan model *pipeline* dan *TfidfTransformer* adalah langkah-langkah yang dapat divalidasi secara bersama dan secara berurutan. Berikut ini adalah kode yang digunakan untuk membuat model pickel

```
X = df.Text
y = df.Skor

text_classifier = Pipeline([('vect', TfidfVectorizer()),
                     ('tfidf', TfidfTransformer()),
                     ('classifier', MultinomialNB(alpha=1.0)
),
                     ])
X_train = np.asarray(X)
text_classifier = text_classifier.fit(X_train, np.asarray(y)
)
```

Model yang diberi nama pipeline dengan menggunakan fungsi pickle kemudian disimpan dan akan digunakan lagi nantinya. Model

pickle disimpan dengan nama model_classifier_nbc.pickle, kode yang digunakan dapat dilihat sebagai berikut:

```
files = open('model_classifier_nbc.pickle', 'wb')
pickle.dump(text_classifier, files)
files.close()

print('Proses Training Naive Bayes Selesai!')
```

Berikut ini kode yang digunakan untuk membuka model pickle yang dapat dilihat sebagai berikut:

```
model = open('model_classifier_nbc.pickle', 'rb')
nbc_classifier = pickle.load(model)
print(nbc_classifier)
```

Proses pickle ini menandakan telah selesainya proses training data menggunakan metode NBC. Fungsi model = open untuk membuka model pickle.

3.2.5 Testing Data

Testing adalah tahapan untuk mengetahui hasil akurasi yang didapatkan dari setiap metode, testing dilakukan untuk menghitung dan memprediksi dari data uji dan data latih. Hal yang pertama dilakukan dalam tahap testing adalah meng-upload data testing yang berjumlah 200 positif dan 200 negatif. Tahapan testing akan menggunakan 2 (dua) metode untuk membandingkan hasil akurasinya.

1. **Metode Naïve Bayes Classifier (NBC)**

   Tahapan dalam *testing* menggunakan metode NBC adalah melakukan pembobotan setiap kata dari data uji dan data latih, yang fungsinya untuk memprediksi kesamaan dalam data uji. NBC adalah salah satu metode probabilitas yang memanfaatkan penghitungannya secara kemungkinan maka dalam tahapan testing perhitungan-perhitungan tersebut akan digunakan. Menggunakan model prediction NBC dapat menghitung kemungkinan akurasinya, model peridiction dapat dilihat sebagai berikut:

```
prediction = nbc_classifier.predict(np.asarray(dataset))
```

```
prediction
```

Fungsi dari model prediction dalam metode NBC yaitu untuk melakukan pelabelan yang secara otomatis. Prediction juga digunakan untuk melakukan perhitungan akurasi dalam testing. Data yang telah melalui tahapan prediction kemudian dibandingkan dengan data yang diberi label secara manual. Hasil dari perbandingan pelabelan dapat dilihat pada Gambar 3. 7

	Unnamed: 0	Teks	Aktual	predic
0	0	aku pernah x ngirim barang pake jne yg pertam...	1	1
1	1	Hai kak Aries mohon maaf untuk informasinya s...	1	1
2	2	Paling best sicepat sama JNE sih jalo di gw	1	1
3	3	kalo mau saya kirim jne aja yaa saya trauma ...	1	1
4	4	JNE Dukung Digitalisasi UMKM Pasca Pandemi	1	1
...	...	...	...	...
195	195	hallo saya ini bertanya Tapi tidak di tanggep...	0	0
196	196	min butuh berapa lama kurir untuk pick up kir...	0	0
197	197	paket jemberbwi hari belum sampe juga udah ke...	0	0
198	198	Lawak banget kurir jnt sicepat di daerah gw ti...	0	0
199	199	Ga cuman jnt kemarin aku si cepat juga gituu	0	1

Gambar 3.7 Hasil perbandingan pelabelan

Fungsi dari perbandingan pelabelan untuk mencari nilai dari TP(*true positif*), FP(*false positif*), TN(*true negative*) dan FN(*true negative*).

2. Metode K-Nearest Neighbor(KNN)

Metode selanjutnya yang digunakan adalah metode KNN. KNN merupakan sebuah algoritma klasifikasi yang melihat data testing dengan jarak terdekat. Adapun kode yang digunakan dalam metode KKN sebagai berikut:

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
Tfidf_vect = TfidfVectorizer()
vector_matrix_train = Tfidf_vect.fit_transform(train_dataset['Text'
])
```

Fungsi library sklearn dengan model CountVectorizer dan TfidfVectorizer yang memiliki fungsi untuk menghitung frekuensi kata yang ada didalam teks. Dalam proses testing data dibutuhkannya fungsi fit_transform untuk menghitung standar data yang nantinya akan diterapkan saat pengujian. Selanjutnya akan dilakukan perhitungan kata menggunakan Tf-IDF kode yang digunakan sebagai berikut:

```
tokens = Tfidf_vect.get_feature_names()
tokens
```

fungsi dari tokens yaitu untuk memanggil hasil dari perhitungan sebelumnya, pada tahapan ini menggunakan model Tf-Idf untuk mengenali kata-kata yang sering muncul dalam dokumen. Hasil dari model Tf-Idf dapat dilihat pada Gambar 3.8

```
'aaaaaaaaaaaa',
'aamiin',
'abang',
'abg',
'abis',
'abisss',
'abistu',
'about',
'acara',
'account',
'acdaafacatcadaxav',
'aceh',
'actually',
'actualnya',
'acuan',
'acungi',
'ad',
'ada',
'adaa',
'adain',
'adakan',
'adalah',
'adanya',
```

Gambar 3.8 Hasil Tf-Idf

Dokumen yang telah dikenali kemudian akan dilakukannya trainig data menggunakan *vector_matrix*, perhitungan training dapat dilihat pada kode berikut:

```
doc_term_matrix_train = vector_matrix_train.todense()
df_train_dataset = pd.DataFrame(doc_term_matrix_train,
                   columns=Tfidf_vect.get_feature_names())
df_train_dataset
```

Fungsi *vector matrix* yaitu untuk menghitung nilai dari TF-idf. Dalam metode KNN proses training dilakukan untuk menghitung jarak kata dari tetangga terdekat. hasil dari perhitungan tf-idf dapat dilihat pada Gambar 3. 9

	___	_stakeholders_	aaaaaaaaaaa	aamiin	abang	abg	abis	abisss	abistu	about	...	yu	yuk	yukk	yuli
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1494	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1495	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1496	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1497	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1498	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0

Gambar 3.9 Hasil *Vector Matrix*

```
from sklearn.metrics.pairwise import cosine_similarity
result=cosine_similarity(vector_matrix_train, vector_matrix_test
)
```

Fungsi yang digunakan menggunakan *library* sklearn dengan model cosine similarity untuk menghitung jarak dari data training ke data testing. *Vector_matrix_train* merupakan hasil perhitungan dari *training* data yang dipanggil saat di awal. Hasil dari pemodelan cosin *similarity* dapat dilihat pada Gambar 3. 10

	0	1	2	3	4	5	6	7	8	9
0	0.027486	0.018958	0.023622	0.021688	0.027213	0.037434	0.055862	0.135979	0.000000	0.000000
1	0.000000	0.093406	0.080790	0.123629	0.000000	0.047901	0.034153	0.000000	0.047530	0.091056
2	0.027446	0.029614	0.036900	0.033877	0.042509	0.027469	0.040993	0.030176	0.000000	0.000000
3	0.063037	0.000000	0.111084	0.077809	0.041063	0.026535	0.039599	0.172980	0.000000	0.000000
4	0.036893	0.000000	0.015110	0.092124	0.000000	0.052968	0.026338	0.019463	0.047705	0.023233
...	...	...	...	...	...	...	...	...	...	...
1494	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1495	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1496	0.031232	0.016164	0.031674	0.000000	0.000000	0.000000	0.028224	0.000000	0.011238	0.000000
1497	0.000000	0.024013	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.017720
1498	0.011092	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.024667

Gambar 3.10 Hasil cosine similarity

Pada Gambar 3. 10 Row 1-1498 adalah data training sedangkan kolom 0-9 adalah data testing. Setiap data akan dibandingkan jaraknya contoh row ke 0 dengan kolom ke 3 memiliki jarak 0.123. Setelah dijumlahkan maka selanjutnya akan dicari nilai maksimalnya, kode yang digunakan dapat dilihat sebagai berikut:

```
Max = df_result.max()
id_max = df_result.idxmax()
print(Max, id_max)
```

Fungsi *result_idmax* untuk mencari nilai data yang paling maksimal. Setelah mendapatkan nilai yang maksimal maka akan dicari letak nilai tersebut. Hasil pencarian dapat dilihat pada Gambar 3. 11

```
0      0.315888
1      0.650538
2      0.407101
3      0.352836
4      0.137091
         ...
195    0.306528
196    0.367907
197    0.306192
198    0.336810
199    0.401746
Length: 200, dtype: float64 0       807
1       746
2       967
3       983
4      1480
       ...
195    1259
196     272
197     546
198     281
199     820
Length: 200, dtype: int64
```

Gambar 3.11 Hasil pencarian nilai