BAB 3
METODE PENELITIAN

Penelitian ini adalah penelitian pemodelan topik dan nanti akan dilakukan analisis sentiment pada data Twitter. Penelitian ini nanti akan memakai algoritma Latent Dirichlet Allocation (LDA) dan Naïve Bayes Classification (NBC). Dari penelitian ini akan di butuhkan data *tweet* yang didapatkan dari Twitter yang berkaitan dengan COVID-19, dan selanjutnya akan dilakukan pengolahan data dengan Jupyter Notebook dan akan di tampilkan dengan aplikasi *web dashboard* berupa data yang sudah benar-benar bersih. Data yang sudah benar-benar bersih tersebut, nantinya akan di pergunakan untuk memetakan informasi dan nilai- nilai sentimen dari netizen di media sosial di Twitter dengan kasus COVID-19 sehingga nantinya didapatkan informasi yang lebih jelas dan sesuai mengenai wabah COVID-19.

Awal dilakukannya penelitian ini adalah dari latar belakang permasalahan yang sudah dibahas, melakukan pemrosesan data yang telah diperoleh dan membentuk pemodelan topik serta melakukan pencarian nilai-nilai sentiment yang maksimal sehingga informasi diperoleh sesuai dengan apa yang diharapakan. Akan di perlukan bahan, alat dan jalannya penelitian untuk melakukan pemodelan topik dan analisis sentimen wabah COVID-19 agar dapat menyelesaikan proses pemodelan topik dan analisis sentimen menggunakan data *tweet*.

3.1 BAHAN PENELITIAN

bahan yang akan dipergunakan pada penelitian ini adalah data *tweet* maupun komentar di media sosisal Twitter yang nantinya berfokus pada wabah Virus Corona (COVID-19).

3.2 ALAT PENELITIAN

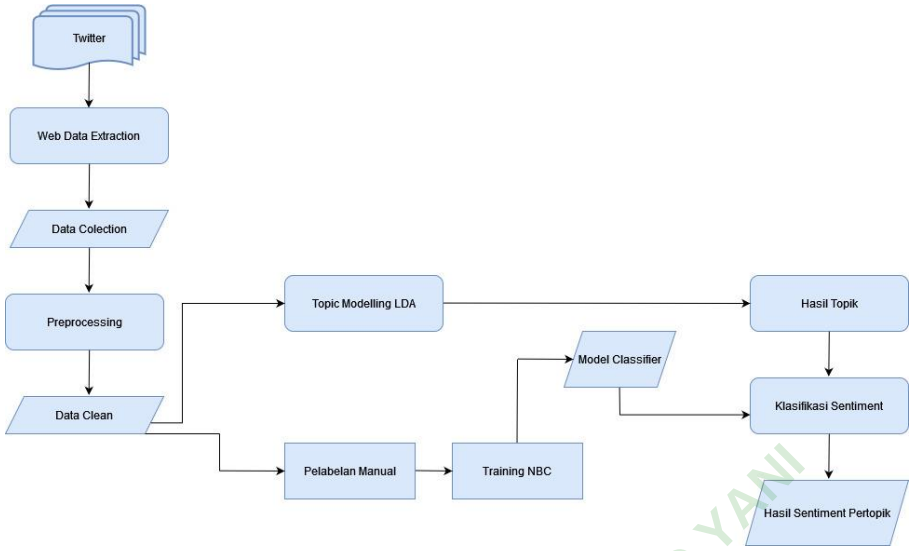
Alat yang akan dipergunakan untuk melakukan penelitian ini adalah leptop denagn kualitas serta spesifikasi yang baik dan cukup untuk melakukan proses pengolahan data serta mampu untuk koneksi ke jaringan internet.

Sistem operasi serta program aplikasi yang akan digunakan dalam pembuatan web dashbord serta pemodelan topik dan analisis sentimen adalah:

1. Sistem Operasi Windows 10 64-bit.
2. Bahasa Pemrograman Python 3.7.3.
3. Microsoft Office Excel 2019.
4. Anaconda 3 64-bit.
5. Jupyter Notebook 6.0.1
6. Sublime text

3.3 JALAN PENELITIAN

Penelitian ini menggunakan Bahasa pemrograman Python, Anaconda 3 dan Jupyter Notebook untuk melakukan pengambilan data yang akan di tampilkan di Microsoft Office Excel dan akan dilakukan *Preprocessing* data, *feature extraction*, dan melakukan pemodelan topik serta proses klasifikasi sentimen dari setiap topik yang sudah didapatkan secara otomatis melalui *website dashboard* yang akan dibuat menggunakan bahasa pemrograman Python. Adapun tahapan pada jalan penelitian ini yaitu:



Gambar 3.1 Alur Penelitian

3.3.1 Web Data Extraction

Tahapan diawali dengan melakukan *web data extraction* dengan mengambil data dan mengumpulkan data *tweet* dari website Twitter mengenai wabah COVID-19 dengan menggunakan Anaconda Promt dan akan diproses di Jupyter Notebook yang akan menghasilkan kumpulan dokumen berbentuk file Microsoft Excel.

3.3.2 Data Collection

Pada tahapan ini data-data yang sudah terkumpulkan dengan 3 kata kunci yaitu: Vaksinasi, Omicron dan delta dengan jumlah data keseluruhan 15.004 dan dilakukan penghilangan *duplicate* data sehingga jumlah data menjadi 10.300 data yang akan digunakan untuk proses analisis dan belum dilakukan *preprocessing* data, dan tersimpan dalam bentuk file Excel, data tersebut dapat dilihat pada Tabel 3.1.

Tabel 3.1 Data Collection

No	Text
1	Peneliti di India menemukan subvarian Omicron BA.2.75 yang merupakan turunan dari BA.2. Kecepatan penyebarannya diperkirakan 9 kali dari varian Delta. https://t.co/DfLomb9rI2
2	Peneliti Kembangkan Rapid Test Identifikasi Varian Delta atau Omicron https://t.co/EZk9kxlM05 https://t.co/nPGS5dJWEF
3	studi terbaru mrk yg mendapatkan vaksin booster pfizer & astrazeneca. peneliti menemukan penerima dua dosis sinovac dgn booster vaksin dari platform vektor (johnson & johnson n astrazeneca) serta mRNA (pfizer n moderna) dpt lebih kuat utk melawan varian seperti delta n omicron. https://t.co/01dhHNNhCK
4	@dutaSherliana Hasil penelitian menunjukkan bahwa korona varian omicron lebih menular daripada Delta #BahanPokokStabil #BBMTerkendali
5	#Paripurna44 Menkeu RI, Sri Mulyani: ...Kasus yang ekstrem, Pemerintah menerapkan kebijakan PPKM Darurat di sebagian besar wilayah NKRI. Untuk merespons dan mengantisipasi dampak varian Delta tersebut, Pemerintah menaikkan alokasi Program PC-PEN menjadi Rp744,8 triliun. (3)
6	@TheMaximvs09 Kalau warga negaranya tetap disiplin prokes dan sudah full vaksin, harusnya ga terlalu ngeri dampaknya.. kayak kemaren omicron aja, menyebar cepat tapi tidak terlalu mematikan severti varian delta, karena sudah banyak yg vaksin
7	@rasyilaa Itu kayaknya lo yg delta ya yg sampe anosmia?gue untungnya ga anosmia sih jadi kemungkinan ini yang varian baru. Thanks yaaaa. Surprise bgt ini org covid udh ilang eh malah gue kena setelah 2 tahun aman aman aja ðŸ¤£
8	@swextdemon Bener-bener sekeluarga sakit, kita bingung mau ngurus bapak gimana karena kita juga sakit banget varian delta gak main-main. Inget banget di ugd saturasi bapakku cuma 59, dan subuhnya bener-bener ngedrop tapi masih ngigo. Tahun 2021 bener-bener berat banget.
9	@hellodeobi Pas thn lalu aku kena juga sama sih kak efeknya sebulanan.. ðŸ˜žini kemungkinan adek kena varian yg delta ditambah sm yg varian baruðŸ˜ž
10	Pada saat ini, hanya Omicron yang masuk daftar varian mengkhawatirkan WHO. Sebelumnya, alfa, beta, gamma dan delta juga termasuk. https://t.co/x37hAzr5uT

Terlihat pada tabel 3.1 terdapat beberapa jumlah data yang masih belum dilakukan *preprocessing* sehingga data tersebut belum dapat digunakan untuk pengolahan data lebih lanjut.

3.3.3 Prepocessing

Setelah melakukan tahapan *web* data *extraction* dilakukan tahapan *prepocessing* data berikut ini tahapan-tahapan pada *preprocessing* data. Sebelum masuk ke tahap *preprocessing* data masukkan *library* yang dapat dilihat pada kode di bawah ini.

```
import numpy as np
import pandas as pd
from pandas import DataFrame
import nltk, emoji, re, string
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

Pada kode diatas digunakan untuk memasukkan *library* serta modul yaitu nltk, emoji, nltk.tokenize, *stopwords* untuk melakukan *preprocessing*

1. **Cleaning**

Tahapan *cleaning* adalah tahapan untuk menghilangkan atribut-atribut yang tidak diperlukan seperti tag, emoji, dan tag url yang dapat dilihat pada kode dibawah ini.

```
 def url_remove(text):
    t = re.sub(r'https://\S+','',text)
    return t

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

def number_remove(text):
    t = re.sub(r"\d+", "", text)
    return t

def regex_remove(text):
    reg = "b',_"
    return re.sub(reg, " ", text)

def emoji_remove(text):
    return emoji.get_emoji_regexp().sub("", text).strip()

def hashtag_remove(text):
    reg = "#(\w+)"
    return re.sub(reg, " ", text)

cleaned = []
```

Sudah terdapat berbagai macam fungsi pada kode diatas dari line pertama untuk menghilangkan url pada data *tweet*, line ke tujuh menghilangkan emoji, dan line delapan ada hastag pada Twitter setelah dilakukan *cleaning* pada data yang tersimpan di dalam data collection tadi data akan terlihat bersih dan dapat dilihat pada Tabel 3.2

Tabel 3.2 Data Cleaning

No	Cleaned_Text
1	hasil teliti corona varian omicron tular delta
2	kayak pas varian delta
3	potensi gelombang subvarian omicronn berat varian delta
4	istilah varian virus covid
5	covid varian delta omicron dari universe
6	varian delta ancam anak muda
7	semenjak covid varian delta tahun yang lalu poko sakit minum obat enggak bisa sembuh cuman istirahat doang
8	varian delta pensiun kah min
9	biar varian omicron sembuh kalau delta bahaya
10	total covid varian omicron indonesia lampau delta

Sudah terlihat pada Tabel 3.2 data yang sudah dilakukan *cleaning* sudah bersih dari url, emoji serta hastag disini data sudah bersih tetapi belum dilakukan proses-proses selanjutnya

2. Tokenizing

Pada tahapan ini berfungsi untuk memisahkan sebuah kata dari kalimat agar menjadi kata tunggal agar kata tersebut dapat berdiri sendiri *library* yang dapat digunakan untuk proses *tokenizing* serta perintah-perintah yang dapat digunakan untuk *tokenizing* dapat dilihat pada kode dibawah ini.

```
def clean_text(text):
    for i in text:

cleaned.append(emoji_remove(punc_remove(number_remove(regex_remov
e(hashtag_remove(url_remove(
            re.sub("[\n\r\t\xa0]"," ",i).strip()))))))
clean_text(df["Text"])
```

Pada kode diatas digunakan untuk memasukkan proses *tokenizing* dan berfungsi untuk memisahkan kata dari kelimat.

3. Casefolding dan Stopwords

Menghilangkan kata-kata yang tidak di perlukan serta atribut-atribut yang sering muncul serta merubah semua huruf menjadi huruf kecil perintah-perintah tersebut dapat di lihat pada kedua kode dibawah ini untuk perintah *casefolding* untuk kode dibawahnya lagi untuk perintah *stopwords*

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

lower_tweet = lowercase()

print(lower_tweet)
```

Dari kode yang sudah terlihat diatas terdapat perintah-perintah untuk merubah kata-kata menjadi huruf kecil.

```
from      Sastrawi.StopWordRemover.StopWordRemoverFactory       import
StopWordRemoverFactory

factory = StopWordRemoverFactory()
more_stopword = ['rt', 'brt', 'anjing', 'anjir', 'anjim', 'pantek',]
stopword = factory.create_stop_word_remover()
stopwords = factory.get_stop_words()+more_stopword
print(stopwords)
```

Dari kode yang terlihat di atas sudah terlihat kata-kata yang tidak perlu lagi digunakan akan di bersihkan dalam proses ini hasilnya dapat dilihat pada Tabel 3.3

Tabel 3.3 Daftar Kata Stopwords

Kata
tempat', 'setengah', 'seterusnya', 'setiap', 'setiba', 'setibanya', 'setidak-tidaknya', 'setidaknya', 'setinggi', 'seusai', 'sewaktu', 'siap', 'siapa', 'siapakah', 'siapapun', 'sini', 'sinilah', 'soal', 'soalnya', 'suatu', 'sudah', 'sudahkah', 'sudahlah', 'supaya', 't', 'tadi', 'tadinya', 'tahu', 'tak', 'tambah', 'tambahnya', 'tampak', 'tampaknya', 'tandas', 'tandasnya', 'tanpa', 'tanya', 'tanyakan', 'tanyanya',

'tapi', 'tegas', 'tegasnya', 'telah', 'tempat', 'tentang', 'tentu', 'tentulah', 'tentunya', 'tepat', 'terakhir', 'terasa', 'terbanyak', 'terdahulu', 'terdapat', 'terdiri', 'terhadap', 'terhadapnya', 'teringat', 'teringat-ingat', 'terjadi', 'terjadilah', 'terjadinya', 'terkira', 'terlalu', 'terlebih', 'terlihat', 'termasuk', 'ternyata', 'tersampaikan', 'tersebut', 'tersebutlah', 'tertentu', 'tertuju', 'terus', 'terutama', 'tetap', 'tetapi', 'tiap', 'tiba', 'tiba-tiba', 'tidak', 'tidakkah', 'tidaklah', 'tiga', 'toh', 'tuju', 'tunjuk', 'turut', 'tutur', 'tuturnya', 'u', 'ucap', 'ucapnya', 'ujar', 'ujarnya', 'umumnya', 'ungkap', 'ungkapnya', 'untuk', 'usah', 'usai', 'v', 'w', 'waduh', 'wah', 'wahai', 'waktunya', 'walau', 'walaupun', 'wong', 'x', 'y', 'ya', 'yaitu', 'yakin', 'yakni', 'yang', 'z', 'rt', 'brt', 'anjing', 'anjir', 'anjim', 'pantek', 'matamu', 'picek', 'Anjj', 'Ndasmu', 'Siluman', 'anjirr', 'tai', 'cen', 'ag', 'man', 'ran', 'fak', 'mff', 'cuy', 'xixi', 'Cipokan', 'bangke', 'Cmiiw', 'tolol', 'Vaxx', 'Hhmmmm', 'anying', 'bego', 'wkwkwkwkwkw', 'euyy', 'iiii', 'uhuy'

Tabel 3.3 di atas adalah daftar kata-kata yang tidak lagi digunakan atau kata-kata yang tidak memiliki makna sehingga perlu dilakukan proses *stopwords*.

4. Stemming

Proses untuk merubah semua kata menjadi kata dasar perintah tersebut dapat dilihat pada kode dibawah ini.

```
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)
```

Kode di atas digunakan untuk melakukan proses *stemming* pada data *tweet* yang sudah didapatkan.

5. Normalization

Proses untuk merubah kata-kata yang disingkat menjadi sebuah kata yang utuh seperti : “yg” menjadi “yang”, “dng” menjadi “dengan” dan lain-lain perintah tersebut dapat dilihat pada kode dibawah ini.

```
normalizad_word = pd.read_excel("normalization.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_word_dict else term for term in document]

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

print(normal_tweet)
```

Dari perintah yang dapat dilihat di Gambar 3.7 ada sebuah file yang berbentuk file Excel file tersebut berisi sebuah kata-kata yang perlu di normalisasikan kata-kata tersebut dapat dilihat pada Tabel 3.4

Tabel 3.4 Tabel Kata Normalisasi

Sebelum	Sesudah
gk	tidak
gua	saya
pake	pakai
bgt	banget
liat	lihat
utk	untuk
yg	yang
jg	juga
krn	karena
kpd	kepada

3.3.4 Data Clean

Tahapan dimana data sudah benar-benar bersih dan sudah dapat digunakan untuk proses selanjutnya perintah untuk menyimpan data *clean* yang sudah benar-benar bersih tersebut dapat dilihat pada kode dibawah ini dan kata-kata tersebut dapat dilihat pada Tabel 3.5.

```
tt={"Text":df['Text'],"Username":df['Username'],"Cleaned_Text":da
ta_preprocess}

df=pd.DataFrame(tt)
df.to_excel("hasil_preprocessing_penggabungan_datafix11k2.xlsx")
```

Pada perintah di atas hasil dari data yang sudah di *preprocessing* akan di simpan atau di unduh dalam bentuk file Excel dan hasilnya dapat dilihat pada Tabel 3.5.

Tabel 3.5 Daftar Data Clean

No	Cleaned_Text
1	hasil teliti corona varian omicron tular delta
2	kayak pas varian delta
3	potensi gelombang subvarian omicronn berat varian delta
4	istilah varian virus covid
5	covid varian delta omicron dari universe
6	varian delta ancam anak muda
7	semenjak covid varian delta tahun yang lalu poko sakit minum obat enggak bisa sembuh cuman istirahat doang
8	varian delta pensiun kah min
9	biar varian omicron sembuh kalau delta bahaya
10	total covid varian omicron indonesia lampau delta

3.3.5 Topic Modelling Menggunakan LDA

Setelah melakukan *preprocessing* pada data akan dilakukan pemodelan topik menggunakan algoritma LDA dengan *library* yang dapat dilihat pada kode kode di bawah ini.

```
        NUM_TOPICS = 30
ldamodel = gensim.models.ldamodel.LdaModel(corpus, num_topics =
NUM_TOPICS, id2word=dictionary, passes=30)
ldamodel.save('modelLDA_30.gensim')

topics = ldamodel.show_topics(num_topics=30,num_words=250)
for topic in topics:
    print(topic)
```

Pada kode diatas memasukkan *library* gensim, lalu menyimpan modelLDA ke dalam file `modelLDA 30.gensim`, serta menampilkan `model LDA`.

```
x=ldamodel.show_topics(num_topics=30,
num_words=250,formatted=False)
topics_words = [(tp[0], [wd[0] for wd in tp[1]]) for tp in x]

#Below Code Prints Topics and Words
for topic,words in topics_words:
    print(str(topic)+ "::"+ str(words))
print()

#Below Code Prints Only Words
topik=[]
for topic,words in topics_words:
     topik.append(" ".join(words))
```

Pada kode diatas untuk menampilkan topik yang telah didapatkan dari perulangan dari "`for topic,words in topics_words:`" dan "`print(str(topic)+ "::"+ str(words))`"

1. **Visualisasi**

Topik-topik yang sudah terkumpulkan yang telah didapatkan dari tahapan *topic modelling* akan di visualisasikan dengan *library*, dan file yang telah tersimpan, serta perintah yang dapat dilihat pada kode dibawah ini

```
dictionary=
gensim.corpora.Dictionary.load('dictionary_lda.gensim')
corpus = pickle.load(open('corpus_lda.pkl', 'rb'))
lda = gensim.models.ldamodel.LdaModel.load('modelLDA_5.gensim')
import pyLDAvis.gensim_models
lda_display = pyLDAvis.gensim_models.prepare(lda, corpus,
dictionary, sort_topics=True)
pyLDAvis.display(lda_display)
```

Pada kode ditas dipergunakan untuk memasukkan modul *library* `pyLDAvis.gensim`, serta memuat file yaitu ada `dictionary_lda.gensim`,

`corpus_lda.pkl` dan `modelLDA_5.gensim`, akan ditampilkan *topic modelling* dari file yang telah dimuat.

Selain itu, topik dapat divisualisasikan dengan kata-kata dalam topik, di mana ukuran setiap kata akan menampilkan frekuensi atau pentingnya kata dalam topik. Sebelum memasuki *library* dan perintah yang dapat dilihat di kode-kode dibawah ini.

```
from os import path
from PIL import Image
from    Sastrawi.StopWordRemover.StopWordRemoverFactory    import
StopWordRemoverFactory, StopWordRemover, ArrayDictionary

import matplotlib.pyplot as plt
```

Pada kode diatas akan digunakan untuk memasukkan modul *library* path, *image, wordCloud, stopwords, imagecolorgenerator*, serta matplotlib.pyplot

```
my_list=tops
wordcloud   =   WordCloud(width   =   1000,   height   =   600,
background_color="white").generate(my_list)
plt.figure(figsize=(15,8))
plt.imshow(wordcloud)
plt.axis("off")
plt.savefig("wordcloud_all"+".png", bbox_inches='tight')
plt.show()
plt.close()
```

Pada kode diatas dipergunakan untuk menampilkan *wordcloud* dari keseluruhan topik serta menyimpanya dengan nama file *wordcloud_all*.png.

```
for data in topik:
    my_list=data
    wordcloud   =   WordCloud(width   =   1000,   height   =   500,
background_color="white").generate(my_list)
    plt.figure(figsize=(10,10))
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.savefig("your_file_name"+".png", bbox_inches='tight')
    plt.show()
    plt.close()
```

Pada kode diatas dipergunakan untuk menampilkan *wordcloud* dari setiap masing-masing topik serta menyimpannya dengan nama file *your_file_name*.png.

2. Analisis

Pada tahap akhir, kualitas topik yang dihasilkan dari *topic modelling* diuji dengan menggunakan hasil *topic cohorence* yang disajikan dalam grafik. Berikut ini adalah beberapa perintah dan library yang digunakan, seperti terlihat pada kode-kode dibawah ini.

```
ldatopics = ldamodel.show_topics(formatted=False)
```

Pada kode diatas tujuannya untuk membuat variable `ldatopics` dan memanggil topik dari proses *topic modelling*.

```
from gensim.models import CoherenceModel, LdaModel, LsiModel, HdpModel
def evaluate_graph(dictionary, corpus, texts, limit):
    """
    Function to display num_topics - LDA graph using c_v coherence

    Parameters:
    ----------
    dictionary : Gensim dictionary
    corpus : Gensim corpus
    limit : topic limit

    Returns:
    -------
    lm_list : List of LDA topic models
    c_v : Coherence values corresponding to the LDA model with respective number of topics
    """
    c_v = []
    lm_list = []
    for num_topics in range(1, limit):
        lm = LdaModel(corpus=corpus, num_topics=num_topics, id2word=dictionary)
        lm_list.append(lm)
        cm = CoherenceModel(model=lm, texts=texts, dictionary=dictionary, coherence='c_v')
        c_v.append(cm.get_coherence())

    x = range(1, limit)
    plt.plot(x, c_v)
    plt.xlabel("num_topics")
    plt.ylabel("Coherence score")
    plt.legend(("c_v"), loc='best')
    plt.show()

    return lm_list, c_v
```

Pada kode diatas dipergunakan untuk memasukkan *library* `CohorenceModel`, `LdaModel`, `LsiModel`, `HdpModel` serta membentuk fungsi untuk memproses *topic cohorence* dari "`def evaluate_graph(dictionary, corpus, texts, limit):`" sampai "`return lm_list, c_v`".

```
%%time
lmlist, c_v = evaluate_graph(dictionary=dictionary, corpus=corpus,
texts=text_data, limit=30)
```

Pada kode diatas dibuat untuk menampilkan grafik dari *topic coherence* sehingga dapat diketahui jumlah topik idealnya dengan jumlah grafik tertinggi.

3.3.6 Naïve Bayes Classifier

Setelah melakukan pemodelan topik menggunakan algoritma LDA selanjutnya akan dilakukan pengolahan data menggunakan metode Naïve Bayes Classifier (NBC) yang akan melewati beberapa tahap seperti pelabelan manual, *feature extraction* TF-IDF, *training*, *testing*, dan klasifikasi data keseluruhan.

3.3.7 Pelabelan Manual

Pelabelan manual adalah proses memberikan label terhadap katimat dan kata yang terdapat pada sebuah dokumen sehingga nantinya dapat dilakukan analisis lebih lanjut mengenai sifat dari kalimat atau kata tersebut apakah memiliki sifat positif atau negatif. Dari data *tweet* yang sudah dilabeli pada tahap ini maka telah di dapatkan data *training* sebanyak 1017 data *tweet* dengan jumlah 518 data *tweet* positif dan 499 data *tweet* negatif. Hasil dari pelabelan manual dapat dilihat pada Gambar 3.2.

	No	Cleaned_Text	label	kelas
0	3	tenlivoyez kasih saran pake parfum delta varia...	negatif	0
1	23	did guys notice suara sirene ambulans frekuens...	negatif	0
2	29	swextdemon aa sorry for your loss kak jd ingat...	negatif	0
3	60	listy vaksin gejala flu doang coba kalau varia...	negatif	0
4	69	pansos status pandemi kampanye harap influence...	negatif	0
...	...	...	...	...
1012	10145	dapat belanja dapat vaksinasi covid gratis loh...	positif	1
1013	10192	duduk besar dunia gt juta persentase cakup vak...	positif	1
1014	10208	monitoring giat vaksinasi covid singkawang gra...	positif	1
1015	10265	pacpeers tarik project data topik analisis pro...	positif	1
1016	10266	covid world vaccination progress analysis prog...	positif	1

1017 rows × 4 columns

Gambar 3.2 Data Pelabelan Manual

Contoh data yang sudah diberikan label manual dapat dilihat pada Tabel

Tabel 3.6 Pelabelan Manual

No	Cleaned_text	Label	Kelas
1	kemarin dokter rs beda bilang ct sudah enggak acu enggak gejala gitu varian delta sih banyakin istirahat buat neng makan bebas enggak pantang makan buat neng saja anw get	positif	1
2	alhamdulillah dulur berkat dukung kerjasama utama masyarakat lamongan yuhronur yes pilih salah kepala daerah jawa timur hasil lewat hantam gelombang covid varian delta	positif	1
3	yaampun semoga cuman pulih saja yak tidak minum vitamin jgn sampai kendor kakk varian delta hamdalah tidak after effect	positif	1
4	cpt sembuh kaa pas varian delta saja udah cpe banget kk	positif	1
5	varian original varian delta varian omicron turun ngapain	positif	1
6	delta hoax varian babi wkwkwk	negatif	0

7	luhut sebut varian omicron parah flu banding omicron varian covid delta	negatif	0
8	gejala omicron parah varian delta potensi alami sakit parah mati lansia vaksin milik komorbid gejala omicron umum demam batuk flu sakit tenggorok	negatif	0
9	suara sirene ambulans frekuensi kayak tahun pas varian delta	negatif	0
10	prediksi puncak omicron ba ba indonesia disebutsebut parah gelombang varian delta omicron sih alas menkes	negatif	0

Dari Tabel 3.6 di atas terdapat informasi bahwa label positif diberi kelas dengan nilai 1 sedangkan untuk label negatif diberi kelas dengan nilai 0. Pelabelan manual ini dilakukan untuk memberi nilai sentimen terhadap kelas positif maupun negatif yang nantinya akan dihitung nilai akurasinya.

3.3.8 Data Training

Proses data *training* dimulai dengan ekstraksi pada data teks menggunakan TF-IDF, pada tahap ini digunakan pendekatan Naive Bayes Classifier, dan dilanjutkan dengan proses *training* data untuk membentuk model klasifikasi yang di simpan dengan format pickle yang nantinya dapat dipergunakan untuk mengklasifikasikan sentimen data keseluruhan secara otomatis. Contoh penghitungan TF-IDF secara manual dengan Microsoft Office Excel ditunjukkan pada Tabel 3.7 di bawah ini.

Tabel 3.7 Dokumen TF-IDF

Dokumen (d)	Kalimat
d1	pasien papar covid varian omicron manggarai tambah
d2	harap masyarakat taat prokes cegah penyeberan varian omicron utama jaga sehat
d3	menkes negara alami lonjak akibat varian omicron level antibodi turun indonesia alami gencar vaksinasi september level antibodi tinggi hajar omicron
d4	disiplin prokes cegah sebar varian omicron utama jaga sehat

Pada Tabel 3.7 terdapat beberapa kata yang nantinya akan digunakan untuk menghitung TF-IDF secara manual dengan jumlah 4 dokumen yaitu d1, d2, d3, d4 melakukan perhitungan TF ini akan digunakan beberapa komponen seperti *term* atau kata dan d adalah jumlah dari komponen data yang akan digunakan yaitu d1, d2, d3 dan d4 dan terdapat df untuk melakukan perhitungan jumlah *term* yang nantinya muncul dari setiap dokumen. Contoh dari perhitungan TF itu sendiri dapat dilihat pada Tabel 3.8.

Tabel 3.8 Perhitungan pada TF

***Term* (kata)**	**d1**	**d2**	**d3**	**d4**	**df**
pasien	1				1
papar	1				1
varian	1	1	1	1	4
omicron	1	1	2	1	5
manggarai	1				1
tambah	1				1
harap		1			1
masyarakat		1			1
taat		1			1
prokes		1		1	2
cegah		1		1	2
penyebaran		1			1
utama		1			1
jaga		1		1	2
sehat		1		1	2
menkes			1		1
negara			1		1
alami			2		2
lonjak			1		1
akibat			1		1
level			2		2

antibodi			2		2
turun			1		1
indonesia			1		1
gencar			1		1
vaksinasi			1		1
september			1		1
tinggi			1		1
hajar			1		1
disiplin				1	1
sebar				1	1

Pada Tabel 3.8 memberikan penjelasan terhadap kemunculan *term* atau kata didalam sebuah kalimat yang telah dimasukan. Pada perhitngan IDF ini akan menggunakan beberapa komponen seperti *term* atau kata, serta tf dan idf yang terhubung dari kesedian suatu *term* dari semua dokumen, di hitung menggunakan N yaitu jumlah dokumen, dan berikut ini adalah contoh perhitungan IDF yang dapat dilihat pada Tabel 3.9.

Tabel 3.9 Perhitungan Pada IDF

***Term* (Kata)**	**df**	**idf**	**idf (N=4)**	**idf (N=1000)**
pasien	1	1	0.60206	3
papar	1	1	0.60206	3
varian	4	0.25	0	2.397940009
omicron	5	0.2	-0.09691	2.301029996
manggarai	1	1	0.60206	3
tambah	1	1	0.60206	3
harap	1	1	0.60206	3
masyarakat	1	1	0.60206	3
taat	1	1	0.60206	3
prokes	2	0.5	0.30103	2.698970004
cegah	2	0.5	0.30103	2.698970004
penyebaran	1	1	0.60206	3

utama	1	1	0.60206	3
jaga	2	0.5	0.30103	2.698970004
sehat	2	0.5	0.30103	2.698970004
menkes	1	1	0.60206	3
negara	1	1	0.60206	3
alami	2	0.5	0.30103	2.698970004
lonjak	1	1	0.60206	3
akibat	1	1	0.60206	3
level	2	0.5	0.30103	2.698970004
antibodi	2	0.5	0.30103	2.698970004
turun	1	1	0.60206	3
indonesia	1	1	0.60206	3
gencar	1	1	0.60206	3
vaksinasi	1	1	0.60206	3
september	1	1	0.60206	3
tinggi	1	1	0.60206	3
hajar	1	1	0.60206	3
disiplin	1	1	0.60206	3
sebar	1	1	0.60206	3
pasien	1	1	0.60206	3
papar	1	1	0.60206	3

Pada Tabel 3.9 memperlihatkan perhitungan IDF secara manual dengan rumus pada persamaan (1) sedangkan untuk penjelasan perhitungan TF-IDF dapat dilihat pada Tabel 3.10.

Tabel 3.10 Perhitungan Pada TF-IDF

***Term* (kata)**	**d1**	**d2**	**d3**	**d4**
pasien	0.6021			
papar	0.6021			
varian	0	0	0	0
omicron	-0.097	-0.097	-0.194	-0.097

manggarai	0.6021			
tambah	0.6021			
harap	0.6021			
masyarakat		0.6021		
taat		0.6021		
prokes		0.301		0.301
cegah		0.301		0.301
penyebaran		0.6021		
utama		0.6021		
jaga		0.301		0.301
sehat		0.301		0.301
menkes			0.6021	
negara			0.6021	
alami			0.602	
lonjak			0.6021	
akibat			0.6021	
level			0.602	
antibodi			0.602	
turun			0.6021	
indonesia			0.6021	
gencar			0.6021	
vaksinasi			0.6021	
september			0.6021	
tinggi			0.6021	
hajar			0.6021	
disiplin				0.6021
sebar				0.6021

Pada Tabel 3.10 menjelaskan perhitungan manual pada TF-IDF secara manual menggunakan Microsoft Office Excel awal dari hasil perkalian tf dan idf.

Klasifikasi pada penelitian ini menggunakan fitur ekstraksi TF-IDF yang memberikan hasil perhitungan secara otomatis pada pembobotan kata pada setiap

dokumen. *Library* yang digunakan untuk perhitungan TF-IDF ini adalah *sklearn.feature_extraction.text* serta *TfidfVectorizer* untuk melakukan perhitungan secara otomatis. Dibantu dengan *library Multinomial* Naïve Bayes dimana akan membantu untuk mengklasifikasi teks pada sebuah data di data *training*. Dan berikut adalah kode utnuk perhitungan TF-IDF didalam sebuah sistem pada kode-kode dibawah ini.

```
from sklearn.feature_extraction.text import TfidfVectorizer

s1 = "vaksin gejala flu doang coba kalau varian delta kemarin lewat
"
s2 = "delta hoax varian babi wkwkwk"

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

X.toarray()
```

```
[(0.0, 'babi'),
 (0.3331023155662109, 'coba'),
 (0.23700504099641823, 'delta'),
 (0.3331023155662109, 'doang'),
 (0.3331023155662109, 'flu'),
 (0.3331023155662109, 'gejala'),
 (0.0, 'hoax'),
 (0.3331023155662109, 'kalau'),
 (0.3331023155662109, 'kemarin'),
 (0.3331023155662109, 'lewat'),
 (0.3331023155662109, 'vaksin'),
 (0.23700504099641823, 'varian'),
 (0.0, 'wkwkwk')]
```


Gambar 3.3 Hasil Perhitungan Pertama

```
[(0.4992213265230509, 'babi'),
 (0.0, 'coba'),
 (0.35520008546852583, 'delta'),
 (0.0, 'doang'),
 (0.0, 'flu'),
 (0.0, 'gejala'),
 (0.4992213265230509, 'hoax'),
 (0.0, 'kalau'),
 (0.0, 'kemarin'),
 (0.0, 'lewat'),
 (0.0, 'vaksin'),
 (0.35520008546852583, 'varian'),
 (0.4992213265230509, 'wkwkwk')]
```

Gambar 3.4 Hasil Perhitungan Kedua

Setelah dilakukan perhitungan TF-IDF secara otomatis oleh sistem selanjutnya akan dilakukan pencarian akurasi dari data *training* yang sebelumnya sudah dilakukan pelabelan manual tujuannya adalah untuk mengetahui keakuratan dari sebuah dokumen atau data tersebut. Kode yang digunakan untuk mencari nilai dari akurasi pada data *training* dapat dilihat pada kode dibawah ini.

```
from sklearn.metrics import accuracy_score, f1_score,
confusion_matrix

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

Cros-validation adalah metode untuk mendapatkan hasil akurasi yang dihitung sampai beberapa kali dengan para meter yang sama. Untuk melakukan pengujian akurasi metode Naive Bayes agar dapat diketahui akurasinya. Konsep pada cross-validation ini adalah membagi dua data yaitu data latih dan data uji Dengan memakai *library from* `sklearn.model_selection` serta *import* `ShuffleSplit` untuk melakukan perhitungan rata-rata dalam 10 kali. Berikut kode untuk menghitung cross validation dapat dilihat pada kode dibawah ini.

```
fig, (ax1, ax2) = plt.subplots(2, 1, sharex=True, figsize=(16,9))

acc_scores = [round(a * 100, 1) for a in accs]
f1_scores = [round(f * 100, 2) for f in f1s]
```

```
x1 = np.arange(len(acc_scores))
x2 = np.arange(len(f1_scores))

ax1.bar(x1, acc_scores)
ax2.bar(x2, f1_scores, color='#559ebf')

# Place values on top of bars
for i, v in enumerate(list(zip(acc_scores, f1_scores))):
    ax1.text(i - 0.25, v[0] + 2, str(v[0]) + '%')
    ax2.text(i - 0.25, v[1] + 2, str(v[1]))

ax1.set_ylabel('Accuracy (%)')
ax1.set_title('Naive Bayes')
ax1.set_ylim([0, 100])

ax2.set_ylabel('F1 Score')
ax2.set_xlabel('Runs')
ax2.set_ylim([0, 100])

sns.despine(bottom=True, left=True)  # Remove the ticks on axes for
cleaner presentation

plt.show()
```

Tahap selanjutnya adalah langkah untuk melakukan perhitungan keakuratan pemodelan pada tahapan *training* yang akan digunakan untuk memprediksi label (kelas) dari data yang sudah tersedia. Dilanjutkan dengan melakukan pembuatan model klasifikasi dengan variabel X serta variabel Y dengan data *training* yang sudah di proses di sebelumnya. Model tersebut dibuatkan sebuah fungsi agar pada saat proses pemanggilannya lebih mudah saat dijalankan untuk tahap berikutnya, sehingga akan lebih efektif dan efisien.

Di dalam proses pembentukan model klasifikasi digunakan *library* *sklearn.pipeline* dengan *import pipeline* yang fungsinya *testable* pada proses cross-validation, lalu *import* pickle memiliki fungsi untuk menyimpan serta membaca file berformat .pkl. Dan berikut kode untuk *import library* pembuatan model klasifikasi terdapat pada kode dibawah ini.

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

Dan untuk pembuatan model klasifikasi dapat dilihat pada kode dibawah ini.

```
X = df.Cleaned_Text
y = df.kelas

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 diberikan nama `text_classifier` akan tersimpan ke dalam bentuk file.pickle sehingga nantinya dapat dibuka serta digunakan kembali. Berikut kode untuk menyimpan file pickle serta membuka file pickle dapat dilihat pada kode-kode dibawah ini.

```
Save pickle

files = open('model_classifier_nbc2.pickle', 'wb')
pickle.dump(text_classifier, files)
files.close()


membuka model klasifikasi
model = open('model_classifier_nbc2.pickle', 'rb')
nbc_classifier = pickle.load(model)
print(nbc_classifier)
```

Bentuk file pickle yang sudah dibentuk model klasifikasi selanjutnya akan dipergunakan untuk menjalankan data *testing* yang mana data *testing* dengan jumlah 425 *tweet* yang sudah dilabeli secara manual dari jumlah data *training* yang berjumlah 1017 *tweet* serta 425 *tweet* data *testing* dari total data 10.300 *tweet* yang sudah dilakukan *filtering* data di Microsoft Office Excel. Dan berikut kode untuk pemanggilan dataframe terdapat pada kode dibawah ini.

```
df_tweet = pd.read_excel (r'result_topic_covid_5.xlsx',
sheet_name='Sheet1')
df_tweet=pd.DataFrame(df_tweet)
df_tweet=df_tweet.dropna()
df_tweet.head()
```

Setelah melakukan pemanggilan dataframe maka langkah selanjutnya adalah melakukan prediksi dari metode Naïve Bayes. Kode pemanggilan untuk hasil prediksi dari Naïve Bayes dapat dilihat pada kode dibawah ini.

```
predicted = nbc_classifier.predict(np.asarray(data_tweet))
```

Lalu selanjutnya hasil prediksi yang sudah didapatkan dari Naïve Bayes akan disimpan didalam variabel *result_tweet* dalam sebuah bentuk data *list*. Kode untuk pemanggilan hasil prediksi dari Naïve Bayes terdapat pada kode dibawah ini.

```
result_tweet=[]
for i in range(len(predicted)):
    if(predicted[i]==1):
        sentiment_result='positive'
    elif(predicted[i]==0):
        sentiment_result='negative'
    result_tweet.append({'tweet':df_tweet['tweet'][i],
'topic':df_tweet['topic'][i],'class':predicted[i],
'result_nbc':sentiment_result})
```

3.3.9 Testing

Pada tahapan testing ini untuk menentukan akurasi dari model yang telah dibuat pada tahapan *training*, bertujuan untuk menentukan label atau kelas dari data *testing* yang telah disediakan. Maka akan ditampilkan penggabungan data *testing* yang dilabeli secara manual (*actual*) dan dari metode Naïve Bayes (*predicted*) terdapat pada Gambar dibawah ini.

	Cleaned_Text	actual	predicted
0	alhamdulillah varian delta tingkat	0	0
1	catat personal covid gelombang varian delta ta...	1	1
2	saya varian delta juli tahun beneran tidak ban...	1	0
3	alhamdulillah varian delta tingkat rumah sakit...	1	0
4	teliti india temu subvarian omicron ba turun b...	0	0
...	...	...	...
419	lindung masyarakat covid tingkat beri vaksinas...	1	1
420	dinas sehat dinkes purbalingga jawa tengah jat...	0	0
421	polisi sektor polsek cempaka putih gelar vaksi...	1	1
422	cepat vaksinasi covid wilayah koramil margorej...	1	1
423	iptu sukresno kapolsek donorojo camat donorojo...	1	1

424 rows × 3 columns

Gambar 3.5 Penggabungan Data Testing

Selanjutnya akan dilakukan pencarian nilai perbandingan model dengan *confusion matrix* terdapat *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *True Negative* (TN). Yang nantinya hasil dari prediksi perbandingan tersebut akan dilakukan perhitungan menggunakan Microsoft Office Excel sehingga akan mendapatkan nilai akurasi dari data testing tersebut.

3.3.10 Klasifikasi

Setelah didapatkan nilai akurasi yang baik dari proses *training* dan proses *testing* selanjutnya dilakukan proses klasifikasi data keseluruhan dari proses klasifikasi ini data yang digunakan adalah data dari keseluruhan dengan jumlah 15.004 data yang sudah dilakukan *filtering* di Microsoft Office Excel menjadi 10.300 data bersih yang digunakan, data tersebut sebelumnya sudah di proses kedalam *classification* metode Latent Dirichlet Allocation sehingga mendapatkan data yang sudah terdapat penyebaran topiknya serta nilai *scorenya*.

Selanjutnya data tersebut akan dilakukan pencarian nilai sentimen dari setiap topiknya menggunakan metode Naïve Bayes Classifier sehingga akan menghasilkan nilai sentimen pertopik.