BAB 3
METODE PENELITIAN

Berikut ini adalah bahan, alat, serta tahapan penelitian guna melakukan analisis kesehatan mental pada Twitter menggunakan algoritma *K-Means Clustering*.

3.1 BAHAN DAN ALAT PENELITIAN

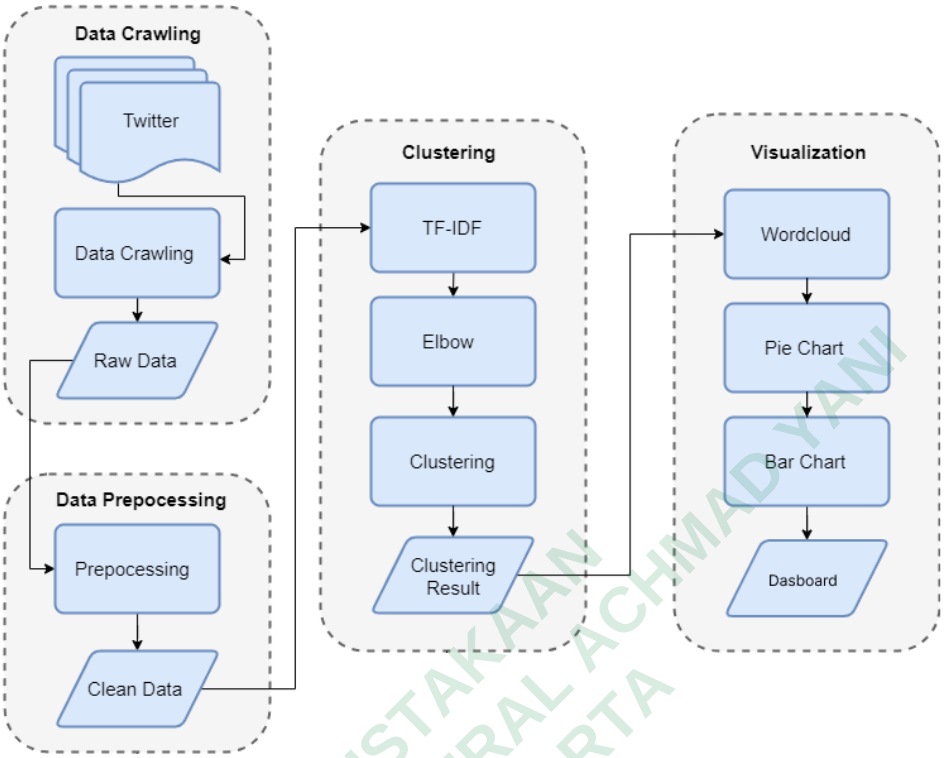
Bahan yang digunakan dalam penelitian ini adalah kumpulan data yang berkaitan dengan kesehatan mental di wilayah Indonesia. Data diperoleh dari media sosial Twitter, dan diolah menggunakan Algoritma *K-Means* dalam bahasa pemrograman Python.

Alat yang digunakan dalam penelitian ini adalah *Mini PC* dengan spesifikasi yang menunjang penelitian serta koneksi internet yang memadai. Adapun sistem operasi dan *software* yang dipergunakan untuk menunjang penelitian ini adalah:

1. Sistem Operasi: Windows 10.
2. Node.js v18.16.0
3. Visual Studio Code 1.75.1.
4. Python 3.11.4.
5. Google Colaboratory.
6. Framework: Django.
7. Twitter

3.2 JALAN PENELITIAN

K-Means clustering adalah salah satu algoritma yang digunakan untuk analisis data. Algoritma *K-Means* dipilih dalam penelitian ini karena kemudahan saat digunakan serta memiliki akurasi yang tinggi dengan meminimalkan variasi data dalam *cluster* dan memaksimalkan data pada *cluster* lainnya. Tahapan yang dilakukan dalam melakukan analisis kesehatan mental pada Twitter menggunakan algoritma *K-Means Clustering* dapat dilihat pada Gambar 3.1.

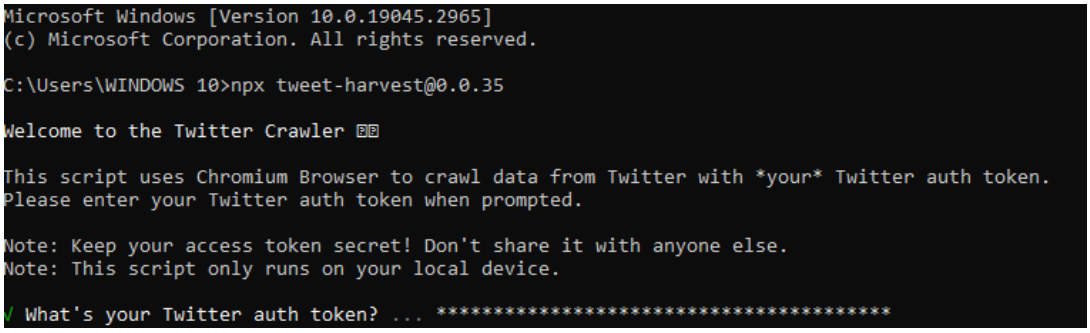


Gambar 3. 1 Tahap Penelitian

Pada gambar diatas, dapat dilihat terdapat banyak tahapan yang dilalui dalam melakukan penelitian. Berikut ini adalah tahap yang dilakukan dalam melakukan analisis kesehatan mental pada Twitter menggunakan algoritma *K-Means Clustering*.

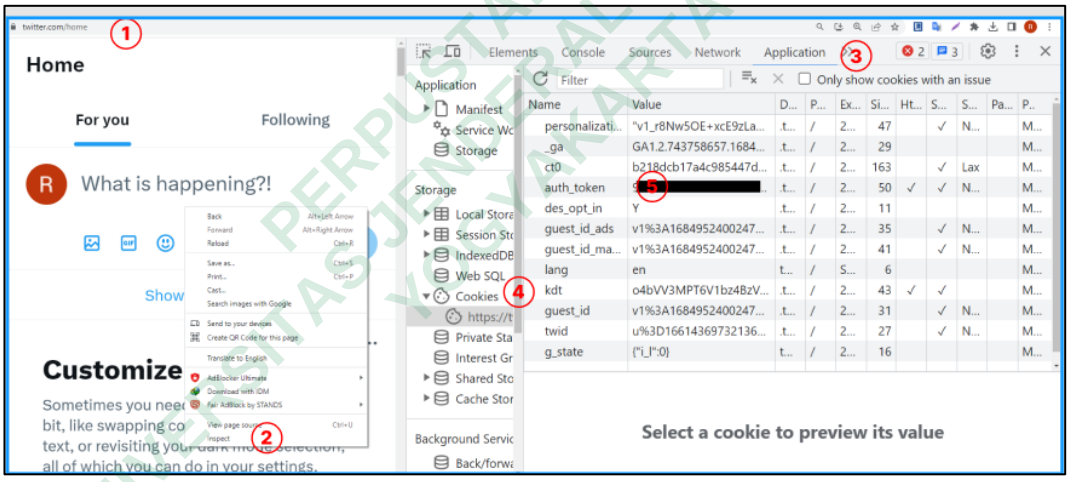
3.2.1 *Data Crawling*

Data Crawling merupakan proses pengambilan data yang berasal dari media sosial dikumpulkan menjadi satu. Dalam penelitian ini data diambil dari media sosial Twitter di karenakan jumlah pengguna yang banyak sehingga data yang diambil memiliki banyak variasi. Node.js dipilih sebagai alat pengambilan data dikarenakan kebijakan baru dari Twitter yang menutup beberapa *moduls data scraping* yang dimiliki Python. Sehingga Node.js digunakan sebagai alternatif dalam melakukan *data scraping*. Penggunaan Node.js dapat dilihat pada Gambar 3.2.



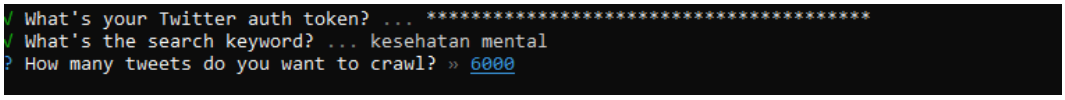
Gambar 3. 2 Install Package tweet-harvest

Langkah awal adalah melakukan instalasi *Package tweet-harvest* yang merupakan *package* Node.js guna melakukan *data* scraping pada twitter. Setelah selesai maka selanjutnya adalah memasukan *auth token* Twitter dengan cara buka Twitter – *inpect* – *application- cookies* dan cari *auth_token.* Langkah pengambilan *auth token* Twitter dapat dilihat pada Gambar 3.3.



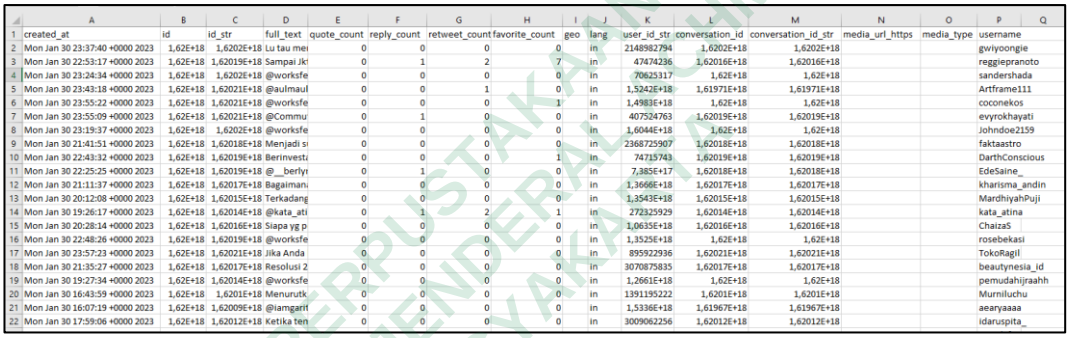
Gambar 3. 3 Pengambilan auth token Twitter

Setelah memasukkan *auth_token* langkah selanjutnya adalah memasukkan *keyword* yang ingin dicari, dalam penelitian ini data yang ingin dicari yaitu "kesehatan mental". Masukkan juga banyak data yang ingin diambil, sebanyak 6000 data yang diambil dari kurun waktu 1 januari 2023 hingga 30 januari 2023. Untuk lebih jelasnya dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Memasukan Keyword dan Value

Data yang diambil belum tentu sebanyak yang diinginkan. Node.js akan memproses data dengan cara membuka Twitter dan melakukan *scrolling* pada pencarian Twitter dengan *keyword* "kesehatan mental" dan akan menyimpan data hasil ke dalam banyak kategori yaitu: *created_at, id, id_str, full_text, quote_count, reply_count, retweet_count, favorite_count, geo, lang, user_id_str, conversation_id, conversation_id_str, media_url_https, media_type,* dan *username*. Data hasil *crawling* dari Node.js dapat dilihat pada Gambar 3.5.

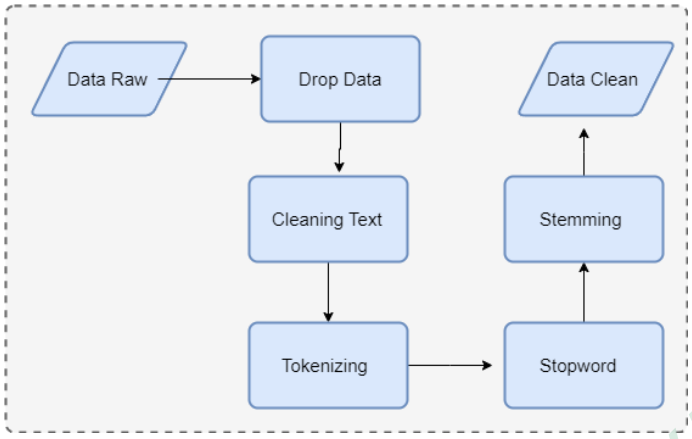


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	created_at	id	id_str	full_text	quote_count	reply_count	retweet_count	favorite_count	geo	lang	user_id_str	conversation_id	conversation_id_str	media_url_https	media_type	username	
2	Mon Jan 30 23:37:40 +0000 2023	1,62E+18	1,6202E+18	Lu tau me	0	0	0	0		in	2148982794	1,6202E+18	1,6202E+18			gwiyoongie	
3	Mon Jan 30 22:53:17 +0000 2023	1,62E+18	1,62019E+18	Sampai Jk	0	1	2	7		in	47474236	1,62016E+18	1,62016E+18			reggiepranoto	
4	Mon Jan 30 23:24:34 +0000 2023	1,62E+18	1,6202E+18	@worksfe	0	0	0	0		in	70625317	1,62E+18	1,62E+18			sandershada	
5	Mon Jan 30 23:43:18 +0000 2023	1,62E+18	1,62021E+18	@aulmaul	0	0	1	0		in	1,5242E+18	1,61971E+18	1,61971E+18			ArtFrame111	
6	Mon Jan 30 23:55:22 +0000 2023	1,62E+18	1,62021E+18	@worksfe	0	0	0	1		in	1,4983E+18	1,62E+18	1,62E+18			coconekos	
7	Mon Jan 30 23:55:09 +0000 2023	1,62E+18	1,62021E+18	@Commu	0	1	0	0		in	407524763	1,62019E+18	1,62019E+18			evyrokhayati	
8	Mon Jan 30 23:19:37 +0000 2023	1,62E+18	1,6202E+18	@worksfe	0	0	0	0		in	1,6044E+18	1,62E+18	1,62E+18			Johndoe2159	
9	Mon Jan 30 21:41:51 +0000 2023	1,62E+18	1,62018E+18	Menjadi si	0	0	0	0		in	2368725907	1,62018E+18	1,62018E+18			faktaastro	
10	Mon Jan 30 22:43:32 +0000 2023	1,62E+18	1,62019E+18	Berinvesti	0	0	0	1		in	74715743	1,62019E+18	1,62019E+18			DarthConscious	
11	Mon Jan 30 22:25:25 +0000 2023	1,62E+18	1,62019E+18	@__berlyr	0	1	0	0		in	7,385E+17	1,62018E+18	1,62018E+18			EdeSaine_	
12	Mon Jan 30 21:11:37 +0000 2023	1,62E+18	1,62017E+18	Bagaiman	0	0	0	0		in	1,3666E+18	1,62017E+18	1,62017E+18			kharisma_andin	
13	Mon Jan 30 20:12:08 +0000 2023	1,62E+18	1,62015E+18	Terkadang	0	0	0	0		in	1,3543E+18	1,62015E+18	1,62015E+18			MardhiyahPuji	
14	Mon Jan 30 19:26:17 +0000 2023	1,62E+18	1,62014E+18	@kata_ati	0	1	2	1		in	27232592	1,62014E+18	1,62014E+18			kata_atina	
15	Mon Jan 30 20:28:14 +0000 2023	1,62E+18	1,62016E+18	Siapa yg p	0	0	0	0		in	1,0635E+18	1,62016E+18	1,62016E+18			ChaizaS	
16	Mon Jan 30 22:48:26 +0000 2023	1,62E+18	1,62019E+18	@worksfe	0	0	0	0		in	1,3525E+18	1,62E+18	1,62E+18			rosebekasi	
17	Mon Jan 30 23:57:23 +0000 2023	1,62E+18	1,62021E+18	Jika Anda	0	0	0	0		in	895922936	1,62021E+18	1,62021E+18			TokoRagil	
18	Mon Jan 30 21:35:27 +0000 2023	1,62E+18	1,62017E+18	Resolusi 2	0	0	0	0		in	3070875835	1,62017E+18	1,62017E+18			beautynesia_id	
19	Mon Jan 30 19:27:34 +0000 2023	1,62E+18	1,62014E+18	@worksfe	0	0	0	0		in	1,2661E+18	1,62E+18	1,62E+18			pemudahijraahh	
20	Mon Jan 30 16:43:59 +0000 2023	1,62E+18	1,6201E+18	Menurutk	0	0	0	0		in	1391195222	1,6201E+18	1,6201E+18			Murniluchu	
21	Mon Jan 30 16:07:19 +0000 2023	1,62E+18	1,62009E+18	@iamgarit	0	0	0	0		in	1,5336E+18	1,61967E+18	1,61967E+18			aearyaaaa	
22	Mon Jan 30 17:59:06 +0000 2023	1,62E+18	1,62012E+18	Ketika ten	0	0	0	0		in	3009062256	1,62012E+18	1,62012E+18			idaruspita_	

Gambar 3. 5 Hasil Data Crawling

3.2.2 *Data Preprocessing*

Data processing merupakan tahap awal dalam melakukan analisis data dimana data akan dibersihkan dan akan disempurnakan semaksimal mungkin guna menjadi data yang berkualitas saat dianalisis. Tahapan *data preprocessing* dapat dilihat pada Gambar 3.6.



Gambar 3. 6 Tahapan Preprocessing

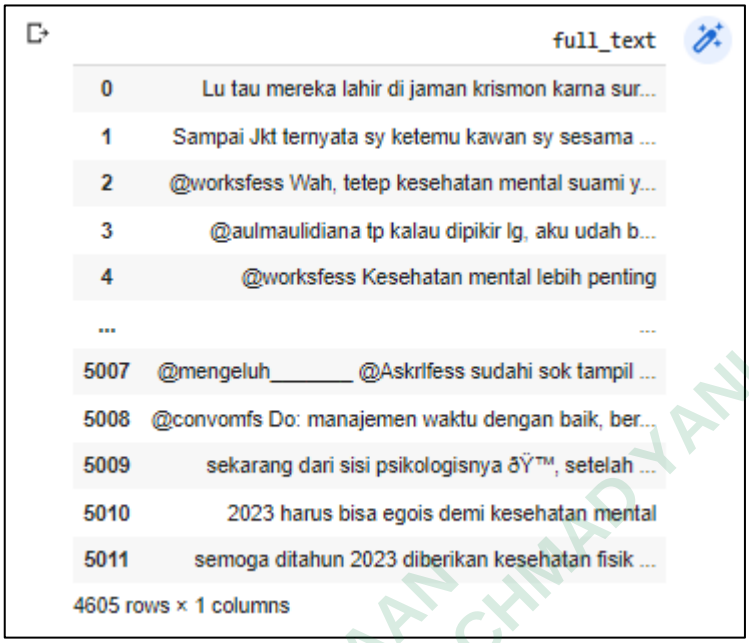
Dari gambar di atas dapat dilihat bahwa *data preprocessing* memiliki beberapa tahapan diantaranya yaitu:

1. *Drop Data*

   Saat setelah melakukan *data crawling*, data yang diperoleh banyak mengandung *column* data yang tidak digunakan sehingga *column* tersebut harus dihapus agar tidak mengganggu jalannya analisis. Selain itu data tentunya tidak luput dengan adanya duplikasi. Baik yang diketik ulang oleh *user*, ataupun dari fitur *retweet*. Data yang sama tentunya akan menurunkan kualitas data yang dimiliki sehingga diperlukan menghapus data yang memiliki duplikasi Berikut adalah *code* dalam melakukan *drop column*:

```
columsdrop = ['created_at', 'id',      'id_str',
              'quote_count','reply_count', 'retweet_count',
              'favorite_count',   'geo', 'lang',      'user_id_str',
              'conversation_id','conversation_id_str',
              'media_url_https', 'media_type', 'username']
data = data.drop(columns=columsdrop)
data.drop_duplicates(inplace=True)
```

   Hasil dari *code* tersebut dapat dilihat pada Gambar 3.7.



	full_text
0	Lu tau mereka lahir di jaman krismon karna sur...
1	Sampai Jkt ternyata sy ketemu kawan sy sesama ...
2	@worksfess Wah, tetep kesehatan mental suami y...
3	@aulmaulidiana tp kalau dipikir lg, aku udah b...
4	@worksfess Kesehatan mental lebih penting
...	...
5007	@mengeluh_____ @Askrlfess sudahi sok tampil ...
5008	@convomfs Do: manajemen waktu dengan baik, ber...
5009	sekarang dari sisi psikologisnya ðŸ™, setelah ...
5010	2023 harus bisa egois demi kesehatan mental
5011	semoga ditahun 2023 diberikan kesehatan fisik ...

4605 rows × 1 columns

Gambar 3. 7 Hasil Drop Data

Dapat dilihat bahwa *column created_at, id, id_str, quote_count, reply_count, retweet_count, favorite_count, geo, lang, user_id_str, conversation_id, conversation_id_str, media_url_https, media_type,* dan *username* telah berhasil dihapus. Selain itu dari *data raw* yang dimiliki terdapat 5012 data sebelum melakukan *drop* duplikasi, dan setelah melakukan *drop* duplikasi data yang dihasilkan menjadi 4605 data

2. *Cleaning Text*

Data mentah yang di ambil dari Twitter menggunakan Node.js tentunya masih sangat banyak kekurangan sehingga perlu dilakukan pembersihan seperti hapus URL yang terdapat dalam *tweet*, hapus tanda @ dalam nama pengguna *(username*), hapus tanda pagar (#) dalam *hashtag* pada *tweet*, hapus angka yang terdapat dalam *tweet*, hapus tanda baca seperti tanda tanya, tanda seru, titik, dan lain-lain. Sehingga perlu dilakukan *cleaning text* agar data yang dihasilkan menjadi lebih bersih dari kata yang tidak berguna. Untuk melakukan itu, maka data harus dibersihkan dengan menggunakan *library* re dengan *code* berikut:

```
def clean_text(text):
    text = re.sub(r'(?:\@|http?\://|https?\://|www)\S+', '', text)
```

```
    text = text.translate(str.maketrans('','',string.punctuation))
    text = re.sub('^a-zA-Z]','',text)
    text = re.sub(r'#\S+', ' ', text)
    text = re.sub(r"b'", ' ', text)
    text = re.sub(r'@[^\s]+', '', text)
    text = re.sub(r'\d+', '', text)
    text = re.sub(r'[^\w\s]+', '', text)
    text = re.sub(r'RT\s+', '', text)
    text = re.sub(r'\\n', ' ', text)
    text = re.sub(r"\b[a-zA-Z]\b","", text)
    text = re.sub('[0-9]+', '', text)
    text = text.lower()
    return text
data['clean'] = [clean_text(i) for i in tweet]
```

Untuk melihat hasil *code* dari *cleaning text* dapat dilihat pada Gambar 3.8.

	full_text	clean
0	Lu tau mereka lahir di jaman krismon karna sur...	lu tau mereka lahir di jaman krismon karna sur...
1	Sampai Jkt ternyata sy ketemu kawan sy sesama ...	sampai jkt ternyata sy ketemu kawan sy sesama ...
2	@worksfess Wah, tetep kesehatan mental suami y...	wah tetep kesehatan mental suami yg utama sih...
3	@aulmaulidiana tp kalau dipikir lg, aku udah b...	tp kalau dipikir lg aku udah bener sih anggap...
4	@worksfess Kesehatan mental lebih penting	kesehatan mental lebih penting
...	...	...
5007	@mengeluh_____ @Askrlfess sudahi sok tampil ...	sudahi sok tampil beda dengan menggurui kaka...
5008	@convomfs Do: manajemen waktu dengan baik, ber...	do manajemen waktu dengan baik bertanggung ja...
5009	sekarang dari sisi psikologisnya ðŸ™, setelah ...	sekarang dari sisi psikologisnya ðÿ setelah se...
5010	2023 harus bisa egois demi kesehatan mental	harus bisa egois demi kesehatan mental
5011	semoga ditahun 2023 diberikan kesehatan fisik ...	semoga ditahun diberikan kesehatan fisik dan ...

4605 rows × 2 columns

Gambar 3. 8 Hasil Cleaning Text

Dapat dilihat dari perbandingan data diatas teks dari data *clean* terlihat lebih bersih daripada teks yang ada di data *full_text*.

3. *Tokenizing*

Tokenizing adalah tahap dimana data dipecah menjadi *token-token* yang lebih kecil agar lebih mudah dalam mengakses dan memanipulasi

bagian-bagian individu dari data. Untuk melakukan *tokenizing* maka diperlukan *library* python nltk, berikut adalah *cod*e untuk melakukan *tokenzing*:

```
from nltk.tokenize import RegexpTokenizer
regexp = RegexpTokenizer(r'\w+|$[0-9]+|\S+')
data['token'] = data['clean'].apply(regexp.tokenize)
```

hasil dari *code* diatas dapat dilihat pada Gambar 3.9.

clean	token
lu tau mereka lahir di jaman krismon karna sur...	[lu, tau, mereka, lahir, di, jaman, krismon, k...
sampai jkt ternyata sy ketemu kawan sy sesama ...	[sampai, jkt, ternyata, sy, ketemu, kawan, sy,...
wah tetep kesehatan mental suami yg utama sih...	[wah, tetep, kesehatan, mental, suami, yg, uta...
tp kalau dipikir lg aku udah bener sih anggap...	[tp, kalau, dipikir, lg, aku, udah, bener, sih...
kesehatan mental lebih penting	[kesehatan, mental, lebih, penting]
...	...
sudahi sok tampil beda dengan menggurui kaka...	[sudahi, sok, tampil, beda, dengan, menggurui,...
do manajemen waktu dengan baik bertanggung ja...	[do, manajemen, waktu, dengan, baik, bertanggu...
sekarang dari sisi psikologisnya ðÿ setelah se...	[sekarang, dari, sisi, psikologisnya, ðÿ, sete...
harus bisa egois demi kesehatan mental	[harus, bisa, egois, demi, kesehatan, mental]
semoga ditahun diberikan kesehatan fisik dan ...	[semoga, ditahun, diberikan, kesehatan, fisik,...

Gambar 3. 9 Hasil Tokenizing

Dari gambar di atas dapat dilihat pada *column token*, data telah dipecah menjadi *token* berdasarkan tiap individu kata. Dengan data yang telah diubah menjadi *token* maka memudahkan melakukan analisis di tahap-tahap selanjutnya.

4. *Stopword*

Stopword adalah Langkah untuk menghapus kata yang tidak memiliki arti ataupun tidak memiliki makna penting. *Stopword* penting untuk dilakukan karena dapat membantu meningkatkan akurasi dalam melakukan analisis teks. *Library* Sastrawi digunakan dalam analisis untuk memudahkan melakukan *stopword* dalam bahasa Indonesia, selain melalui Sastrawi terdapat beberapa kata tambahan yang dibuat secara terpisah agar data lebih bagus saat digunakan. Untuk melakukan *stopword* maka bisa menggunakan *code* berikut:

```
From Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory
factory = StopWordRemoverFactory()
more_stopword = ['lu', 'tau', 'di', 'jkt', 'sy', 'wah', 'yg', 'tp',
'lg', 'do', 'sok', 'amp', 'aja', 'mau', 'bgt', 'ga', 'g', 'y' 'sih',
'nga', 'aku', 'kamu', 'dia', 'mereka', 'orang', 'ok' 'gak']
stopword = factory.create_stop_word_remover()
stopwords = factory.get_stop_words()+more_stopword
def removeStopWords(clean):
    clean_word_list = [word for word in clean.split() if word not
in stopwords]
    return clean_word_list
stopwords_tweet = tweet.apply(removeStopWords)
data['stopwords'] = stopwords_tweet
```

untuk melihat hasil *stopword* dapat dilihat pada Gambar 3.10.

token	stopwords
[lu, tau, mereka, lahir, di, jaman, krismon, k...	[lahir, jaman, krismon, karna, suruh, resign, ...
[sampai, jkt, ternyata, sy, ketemu, kawan, sy,...	[ternyata, ketemu, kawan, sesama, aktivis, kes...
[wah, tetep, kesehatan, mental, suami, yg, uta...	[tetep, kesehatan, mental, suami, utama, sih, ...
[tp, kalau, dipikir, lg, aku, udah, bener, sih...	[kalau, dipikir, udah, bener, sih, anggap, ana...
[kesehatan, mental, lebih, penting]	[kesehatan, mental, lebih, penting]
...	...
[sudahi, sok, tampil, beda, dengan, menggurui,...	[sudahi, tampil, beda, menggurui, kakakkakakmu...
[do, manajemen, waktu, dengan, baik, bertanggu...	[manajemen, waktu, baik, bertanggung, jawab, k...
[sekarang, dari, sisi, psikologisnya, ðÿ, sete...	[sekarang, sisi, psikologisnya, ðÿ, sekian, ta...
[harus, bisa, egois, demi, kesehatan, mental]	[egois, kesehatan, mental]
[semoga, ditahun, diberikan, kesehatan, fisik,...	[semoga, ditahun, diberikan, kesehatan, fisik,...

Gambar 3. 10 Hasil Stopword

Dari data di atas dapat dilihat beberapa kata yang tidak penting telah terhapus pada *column stopword.*

5. *Stemming*

Stemming adalah proses untuk menghilangkan imbuhan kata untuk mengembalikan kata ke bentuk kata dasar. Bertujuan untuk mengurangi variasi kata yang serupa menjadi kata dasar yang sama. Untuk melakukan *stemming* dalam kata Indonesia diperlukan *library* Sastrawi dengan menggunakan *code* berikut:

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory=StemmerFactory()
stemmer = factory.create_stemmer()
def stemmed_wrapper(term):
  return stemmer.stem(term)
term_dict = {}
tweet = data['stopwords']
for doc in tweet:
  for term in doc:
    if term not in term_dict:
      term_dict[term]=''
for term in term_dict:
  term_dict[term] = stemmed_wrapper(term)
def get_stemmed_term(doc):
  return [term_dict[term] for term in doc]
data['tweet'] = data['stopwords'].apply(get_stemmed_term)
```

Untuk melihat hasil *code* di atas maka dapat melihat Gambar 3.11.

stopwords	tweet
[lahir, jaman, krismon, karna, suruh, resign, ...	[lahir, jaman, krismon, karna, suruh, resign, ...
[ternyata, ketemu, kawan, sesama, aktivis, kes...	[nyata, ketemu, kawan, sama, aktivis, kesmen, ...
[tetep, kesehatan, mental, suami, utama, sih, ...	[tetep, sehat, mental, suami, utama, sih, menu...
[kalau, dipikir, udah, bener, sih, anggap, ana...	[kalau, pikir, udah, bener, sih, anggap, anak,...
[kesehatan, mental, lebih, penting]	[sehat, mental, lebih, penting]
...	...

Gambar 3. 11 Hasil Stemming

Dapat dilihat pada gambar di atas terdapat beberapa kata yang telah kembali ke kata dasarnya dan menghilangkan imbuhan kata yang dimilikinya.

3.2.3 *Clustering*

Setelah data dibersihkan dalam tahap *preprocessing*, Langkah selanjutnya adalah melakukan *clustering* data. *Clustering* Merupakan metode pengelompokan data ke dalam himpunan bagian yang disebut dengan *cluster*. *Clustering* membagi suatu populasi atau titik data menjadi beberapa kelompok sehingga titik data pada kelompok yang sama lebih mirip dari titik data lain pada kelompok yang sama dan berbeda dengan titik data pada kelompok lain (Nur dkk., 2017).Terdapat beberapa tahapan dalam melakukan *clustering*. Dapat dilihat pada Gambar 3.1.

Terdapat empat Langkah dalam melakukan *clustering* yaitu:

1. TF-IDF

   TF-IDF merupakan metode yang digunakan dalam pemrosesan teks untuk mengevaluasi pentingnya sebuah kata. Berikut ini merupakan *code* dalam melakukan perhitungan TF-IDF menggunakan *library* sklearn untuk memudahkan dalam perhitungan.

```
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer()
```

```
check=[]
for x in data.tweet:
    check.append(x)
X = vectorizer.fit_transform(data['tweet'].values.astype('U'))
```

Hasil dari *code* di atas dapat dilihat pada Gambar 3.12.

```
  (0, 6140)     0.06853384528438138
  (0, 9522)     0.06683471827653867
  (0, 9092)     0.3556755387134859
  (0, 10380)    0.40687112461639696
  (0, 4664)     0.31941455155692133
  (0, 5188)     0.5338596981930755
  (0, 4272)     0.39324406142839574
  (0, 5329)     0.3963764708555258
  (1, 8743)     0.22062043002255116
  (1, 10782)    0.14724557110877393
```

Gambar 3. 12 Hasil TF-IDF

2. *Elbow*

Elbow bertujuan untuk menentukan jumlah *cluster* yang akan digunakan. Dalam implementasi *code* untuk memudahkan dalam melakukan perhitungan *elbow maka* diperlukan *library* sklearn dan matplotlib untuk memudahkan menampilkan data. *Code* dapat ditulis seperti:

```
Sum_of_squared_distances = []
K = range(1,10)
for k in K:
    km = KMeans(n_clusters= k, init = 'k-means++', random_state =
100)
    km = km.fit(X)
    Sum_of_squared_distances.append(km.inertia_)
plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('Jumlah Cluster')
plt.ylabel('Sum_of_squared_distances')
plt.title('Metode Elbow')
plt.show()
```

3. Clustering Data

Langkah terakhir adalah melakukan *clustering*, data dikelompokkan ke dalam setiap *cluster*. Jumlah *cluster* ditentukan berdasarkan *cluster* paling optimal pada grafik *elbow*. Untuk melakukan *clustering* dapat menggunakan *code* berikut:

```
jk = 4
model = KMeans(n_clusters=jk, init='k-means++', max_iter=200, n_init=10)
model.fit(X)
labels=model.labels_
wiki_cl=pd.DataFrame(list(zip(data.tweet,labels)),columns=['title','cluster'])
print(wiki_cl.sort_values(by=['cluster']))
data['cluster'] = wiki_cl['cluster']
```

hasil *code* di atas dapat dilihat pada Gambar 3.13.

```
                                                  title  cluster
0     ['lahir', 'jaman', 'krismon', 'karna', 'suruh'...        0
2876                      ['sehat', 'mental', 'bang']        0
2877  ['generasi', 'lebih', 'aware', 'sehat', 'menta...        0
2878  ['peduli', 'sama', 'sehat', 'mental', 'cuma', ...        0
2879  ['kafein', 'tingkat', 'energi', 'tingkat', 'ke...        0
...                                                 ...      ...
723   ['putus', 'hubung', 'siapa', 'usaha', 'selalu'...        3
724   ['cari', 'bantu', 'profesional', 'bantu', 'kel...        3
725   ['atas', 'masalah', 'sehat', 'mental', 'sendir...        3
515   ['dewasa', 'diri', 'ndiumur', 'gin', 'sering',...        3
2054  ['kalau', 'sehat', 'mental', 'anggap', 'remeh'...        3

[4605 rows x 2 columns]
```

Gambar 3. 13 Hasil Clustering

Dapat dilihat pada *column* terakhir data telah dimasukkan kedalam variable *cluster* yang sesuai. Setelah melewati tahap *clustering* maka data siap digunakan.

3.2.4 *Visualization*

Setelah data memasuki proses *clustering* dan telah dianggap baik maka tahap selanjutnya adalah mempresentasikan data yang dimiliki agar mudah dimengerti oleh orang lain. Salah satu caranya adalah menggunakan *wordcloud*. *Wordcloud* adalah representasi visual dari kata yang paling sering muncul ditampilkan dengan ukuran yang lebih besar dan lebih menonjol. Berikut merupakan *code* untuk menampilkan *wordcloud*:

```
from wordcloud import WordCloud
result={'cluster':labels,'wiki':data.tweet}
result=pd.DataFrame(result)
for k in range(0,jk):
	s=result[result.cluster==k]
	text=s['wiki'].str.cat(sep=' ')
	text=text.lower()
	text=''.join([word for word in text.split()])
	wordcloud = WordCloud(max_font_size=50, max_words=100,
	background_color="white").generate(text)
	print('Cluster: {}'.format(k))
	plt.figure()
	plt.imshow(wordcloud, interpolation="bilinear")
	plt.axis("off")
	plt.show()
```

Semakin banyak kata yang muncul dalam *wordcloud* maka semakin kompleks topik yang dibahas pada *cluster* tersebut. Selain *wordcloud*, data juga dapat divisualisasikan dalam bentuk lain. *Library* matplotlib dan seaborn menyediakan banyak cara untuk memvisualisasikan data yang dimiliki, contohnya adalah *pie chart* dan *bar chart*. Berikut adalah *code* untuk menampilkan data dengan *pie chart* dan *bar chart:*

```
Pie = [np.count_nonzero(data['cluster'] == 0),
	np.count_nonzero(data['cluster'] == 1),
	np.count_nonzero(data['cluster'] == 2),
	np.count_nonzero(data['cluster'] == 3)]
label = ['Data Cluster 0', 'Data Cluster 1', 'Data Cluster 2', 'Data
Cluster 3']
```

```
plt.title('Perbandingan Data tiap Cluster')
plt.pie(pie, labels = label, autopct='%.1f%%')
plt.show()
y_pos = np.arange(len(objects))
jumlahdata = [np.count_nonzero(data['cluster'] == 0),
np.count_nonzero(data['cluster'] == 1), np.count_nonzero(data['cluster']
== 2), np.count_nonzero(data['cluster'] == 3)]
fig, ax = plt.subplots()
barchart = sns.barplot(x=y_pos, y=jumlahdata, ax=ax)
barchart.bar_label(ax.containers[0])
```

Dengan menampilkan data menggunakan chart maka dapat dengan jelas dilihat *cluster* mana yang memiliki jumlah data terbanyak.