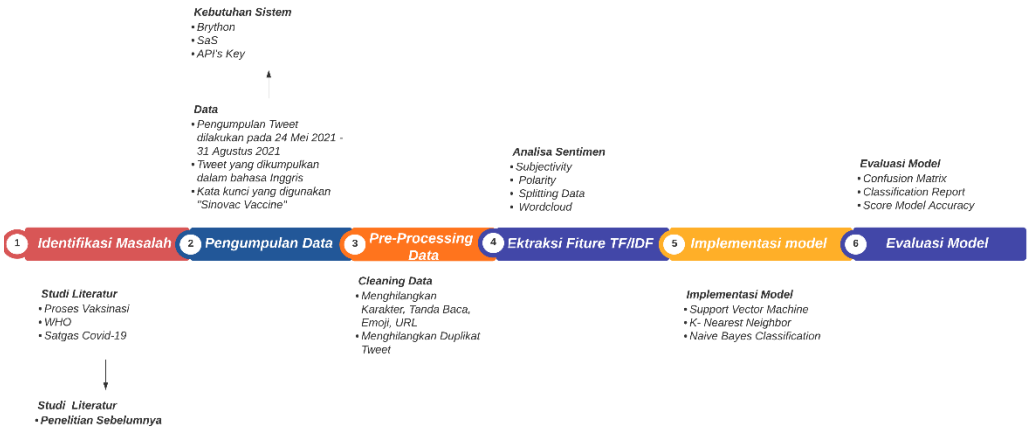
BAB III
METODOLOGI PENELITIAN

Berikut ini gambar 3.1 yang merupakan tahapan penelitian yang dilakukan pada penelitian ini.



Gambar 3. 1 Tahapan Penelitian

3.1 Identifikasi Masalah

Berdasarkan gambar 3.1 tahapan identifikasi masalah dilakukan dalam dua tahapan yaitu:

a. Studi Lapangan

Studi lapangan dilakukan dengan memantau proses vaksinasi di seluruh dunia melalui laman resmi organisasi kesehatan dunia (WHO). Proses vaksinasi di Indonesia dapat diakses melalui laman satgas covid-19 (Covid19.go.id).

Reaksi publik dunia mengenai proses vaksinasi dapat dipantau melalui media sosial khususnya *Twitter*, melalui *platform* ini publik dapat mengeluarkan opini atau pendapat dengan bebas.

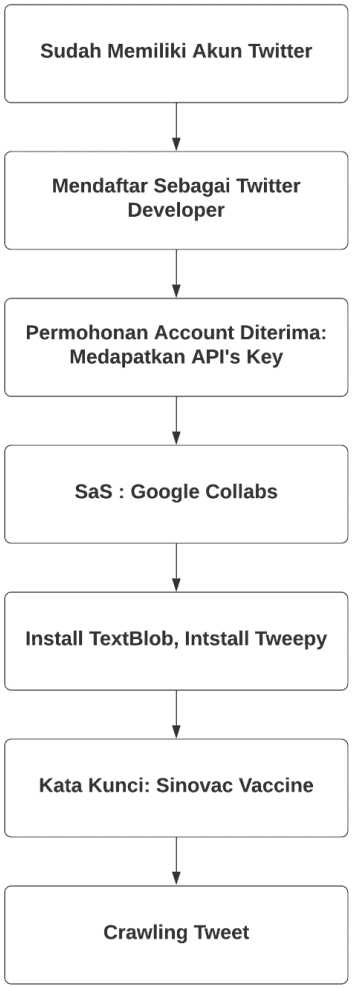
b. Studi Literatur

Studi literatur dilakukan dengan melakukan studi pustaka pada penelitian sebelumnya terkait reaksi publik terhadap proses vaksinasi. Dari hasil

identifikasi masalah diketahui bahwa proses vaksinasi ditanggapi pro dan kontra oleh publik yang diekspresikan melalui media sosial yang salah satunya adalah *Twitter*.

3.2 Pengumpulan Data

Proses pengumpulan data tweet dari Twitter menggunakan API Key dan library Python yaitu TextBlob dan Tweepy seperti pada gambar 3.2 dibawah ini.



Gambar 3. 2 Proses Pengumpulan Data

Berdasarkan gambar 3.2 maka proses pengumpulan data tweet dipaparkan dalam tahapan-tahapan proses dibawah ini:

a. Memiliki Akun Twitter

Data yang akan dianalisa adalah *tweet* dari pengguna *Twitter* dari seluruh dunia, untuk dapat melakukan proses pengumpulan *tweet* / *crawling tweet.* Tahapan pertama yang harus dilakukan adalah memiliki akun sebagai pengguna *Twitter*.

b. Mendaftar Sebagai Twitter Developer

Mengajukan permohonan sebagai *Twitter Developer* untuk mendapatkan *API Key*.

c. Mendapatkan Api Key

Dikabulkannya permohonan sebagai *Twitter Developer* maka *Twitter* akan mengirimkan kode *API Key*. *API key* mempermudah *developer* untuk mengakses informasi pada *web Twitter*.

d. Google Collab

Google collab akan digunakan untuk seluruh proses eksekusi *script* program dalam hal ini pemrograman yang digunakan adalah *Python*.

e. Install TextBlob dan Install Tweepy

TextBlob dan *Tweepy* adalah dua library *Python* yang di dipasang di *Google Collab*. *Tweepy* digunakan untuk akses ke *Twitter API* dan *TextBlob* untuk mencari polaritas.

f. Menggunakan Kata Kunci " Sinovac Vaccine"

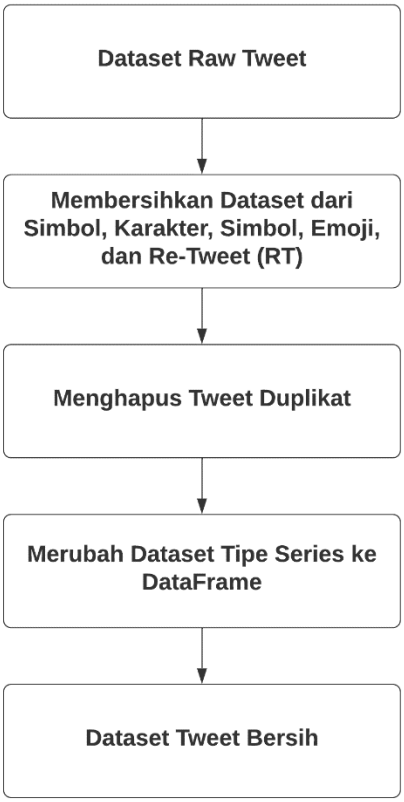
Pengumpulan *tweet* publik terhadap proses vaksinasi menggunakan kata kunci "*Sinovac Vaccine*" dalam bahasa Inggris.

g. Crawling Tweet

Pengumpulan *tweet* atau *crawling tweet* dilakukan pada rentang periode 24 Mei 2021 – 31 Agustus 2021. *Crawling* dilakukan dua jam sekali dan *tweet* yang diperoleh dalam sekali *crawling* adalah 100 baris data dan seluruh *tweet* hasil *crawling* akan di *merge* menjadi satu *file dataset* atau *dataset raw tweet*. Berikut ini tabel proses pengumpulan data *tweet*.

3.3 Pre-processing

Proses *preprocessing* pada gambar 3.3 terdiri dari beberapa tahapan yang memiliki tujuan penting yaitu untuk mendapatkan dataset bersih.



Gambar 3. 3 Proses Pre-Processing

Proses pada gambar 3.3 akan dipaparkan dengan jelas sebagai berikut.

a. Dataset Raw Tweet

Dataset raw tweet yang telah digabungkan atau di-*merge* dalam satu *file dataset* yang diberi nama "*mergetweet"*. *Dataset mergetweet* ini masih merupakan kumpulan *tweet* asli yang masih mengandung simbol-simbol dan karakter-karakter.

b. Menghilangkan Karakter dan Simbol

Dataset "mergetweet" masih mengandung karakter-karakter dan simbol-simbol yang tidak dibutuhkan pada proses selanjutnya. Karakter dan simbol tersebut dihilangkan dengan menjalankan fungsi-fungsi pada tabel 3.1 dibawah ini.

Tabel 3. 1 Menghilangkan Karakter dan Simbol

Fungsi	Keterangan
text = re.sub(r'@[A-Za-z0-9]+', '', text)	Menghilangkan @mentions
text = re.sub(r'[0-9]+', '', text)	Menghilangkan angka
text = re.sub(r'#','',text)	Menghilangkan # hashtag symbol
text = re.sub(r'[^\w]', ' ', text)	Menghilangkan simbol
text = re.sub(r'_', ' ', text)	Menghilangkan underscore
text = re.sub(r'[\n]+', '', text)	Menghilangkan new line
text = re.sub(r':', '', text)	Menghilangkan simbol-simbol
text = re.sub(r'RT[\s]+', '', text)	Menghilangkan RT word
text = re.sub(r'rt[\s]+', '', text)	Menghilangkan RT word
text = re.sub(r't.co','',text)	Menghilangkan # hashtag symbol
t = re.sub(r'https','',text)	Menghilangkan # hashtag symbol
text = re.sub(r'^https?:\/\/\S+', '', text)	Menghilangkan hyperlink https
text = re.sub(r'^ http?:\/\/.[\r\n]*', '', t ext, flags=re.MULTILINE)*	Menghilangkan hyperlink https

c. Menghapus Tweet Duplikat

Selanjutnya adalah menghilangkan duplikasi *tweet* pada *dataset* dengan menjalankan fungsi *drop duplicate* seperti pada *script program* di bawah ini.

```
df4 = df3['Tweet Bersih'].drop_duplicates()
```

d. Merubah Dataset Type Series ke DataFrame

Dataset mergetweet masih dalam *type series* dan akan diubah kedalam *tipe data frame* dengan menjalankan fungsi *script program* berikut ini.

```
pandas.core.series.Series
df5 = df4.to_frame()
```

e. Dataset Tweet Bersih

Tweet duplikat telah dihilangkan pada *dataset mergetweet* maka data *tweet* pada *dataset mergetweet* sudah bersih dari simbol-simbol, karakter-karakter, dan duplikat. *Dataset* bersih akan digunakan pada tahapan berikutnya.

3.4 Ekstraksi Fitur TF/IDF

TF/IDF digunakan dalam ekstraksi fitur dalam sentimen analisis. Ekstraksi fitur dilakukan untuk mengetahui nilai polaritas dan subjektivitas pada *dataset mergetweet*.

a. Polarity

Nilai range polaritas yaitu pada range [-1, 1], semakin pernyataan mendekati nilai 1 maka pernyataan cenderung positif dan sebaliknya semakin mendekati -1 maka pernyataan cenderung negatif.

b. Subjectivity

Nilai subjektivitas pada umumnya mengacu pada pendapat pribadi, emosi atau penilaian. Range nilai untuk subjektivitas adalah [0, 1].

c. Splitting Data

Splitting data atau pembagian *dataset* dibagi ke dalam *data training* dan *data testing*. *Data training* merupakan data yang digunakan untuk melatih model sedangkan *data testing* merupakan data yang digunakan untuk menguji performa suatu algoritma pembelajaran mesin satu dengan yang lainnya [19].

Merujuk pada penelitian sebelumnya [25], pembagian dataset dapat dilakukan dalam beberapa model skenario. Pada penelitian ini akan menggunakan tiga model skenario seperti pada tabel 3.2.

Dalam hal ini pustaka *Python* yang digunakan untuk pembagian *dataset* adalah *Scikit-learn* dengan menjalankan fungsi dibawah ini.

Tabel 3. 2 Pembagian Dataset

No	Fungsi	Keterangan
1	*from sklearn.model_selection import train_test_spl* *X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7)*	Model 1
2	*from sklearn.model_selection import train_test_spl* *X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8)*	Model 2
3	*from sklearn.model_selection import train_test_spl* *X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.9)*	Model 3

Pada tabel 3.2 dapat dijelaskan dimana *x* merupakan *fitur input*, *y* merupakan *label output*, dan *train_size* merupakan berapa persen dari *dataset* yang akan digunakan menjadi *training data*. *X_train* dan *Y_train* merupakan pasangan *fitur input* dan *label output* dari *training data*, sementara *X-test dan Y_test* merupakan pasangan *fitur input* dan *label output* dari *testing data*.

3.5 Implementasi Model

Setelah dilakukan pembagian *data training* dan *data testing*, selanjutnya implementasi model yang dipakai yaitu *Support Vector Machine (SVM)*, *K-Nearest Neighbor (KNN)*, dan *Naive Bayes Classification (NBC)*.

Penelitian terkait klasifikasi sentimen pada Twitter telah dilakukan oleh peneliti sebelumnya [19] menggunakan *SVM* dan *Naive Bayes* untuk klasifikasi sentimen : negatif, positif, dan negatif. Hasil penelitian menunjukkan SVM menghasilkan nilai akurasi lebih baik dibandingkan dengan Naive Bayes dengan nilai akurasi 80% sedangkan Naive bayes 79%. Penelitian [27] menggunakan *Naive Bayes*, *SVM*, *Decision Tree*, dan *KNN* untuk klasifikasi sentimen dan bahasa pemrograman *Python* yang digunakan didalam penelitian tersebut. *SVM* menunjukkan nilai akurasi terbaik dibandingkan dengan metode lain yang digunakan, nilai akurasi SVM setelah dilakukan evaluasi metric mencapai 99.7%. Penelitian lainnya [28]

menggunakan *SVM* dan *multinomialNB* dengan menggunakan 10 *fold-validation*. Dan SVM menghasilkan nilai akurasi terbaik dengan nilai akurasi diatas 90%.

a. Support Vector Machine (SVM)

Pada model SVM terdapat beberapa parameter yang digunakan untuk menghasilkan hyperplane pemisah terbaik, salah satunya adalah parameter C[29] yaitu parameter regulasi SVM untuk semua kernel dengan nilai default = 1.0. Tabel 3.3 merupakan fungsi yang digunakan untuk model *Support Vector Machine*, dengan parameter C dimulai dari C = 0.01 sampai dengan C = 1.

Tabel 3. 3 Support Vector Machine

Fungsi	Keterangan
for c in [0.01, 0.05, 0.25, 0.5, 0.75, 1]: *svm = LinearSVC(C=c)* *svm.fit(X_train, y_train)* *print('Akurasi untuk c = %s: %s' %(c, accuracy_score (y_test, svm.predict(X_test))))*	Nilai default untuk parameter C = 1.0 *Support Vector Machine* Proses training data latih, prediksi data uji, perhitungan akurasi

b. K-Nearest Neighbor

Model kedua adalah *K-Nearest Neighbor (KNN)*, model ini ditentukan oleh kesamaan fitur yaitu seberapa akurat outlier terlihat mirip dengan data pada *data training* dan kesamaan itu menentukan bagaimana titik data uji diklasifikasikan [30].

Tabel 3.4 merupakan fungsi untuk menjalankan model KNN dalam menentukan nilai k secara manual dan tabel 3.5 akan menggunakan metode *Elbow Value* untuk menentukan nilai k.

Tabel 3. 4 K-Nearest Neighbor

Fungsi	Keterangan
knn = KNeighborsClassifier()	*K-NN Classifier*
knn.fit(X_train, y_train)	Proses training data latih,
y_pred_1 = knn.predict(X_test)	prediksi data uji, perhitungan
score = metrics.accuracy_score	akurasi
(y_test, y_pred_1)	
print('Akurasi dengan menggunakan	
K-Nearest Neighbor', score)	

.

Tabel 3.5 untuk menentukan nilai k menggunakan metode *Elbow Value*.

Tabel 3. 5 K-Nearest Neighbor Menggunakan Elbow Value

Fungsi	Keterangan
nilai_k = range(1, 6)	Parameter k = 6
plt.figure(figsize=(10, 6))	Menentukan nilai k menggunakan
plt.plot(nilai_k, error_rate)	Elbow Value
plt.title('Nilai Elbow Values')	
plt.xlabel('Nilai K')	
plt.ylabel('Error Rate')	

c. Naive Bayes Classification

Model ketiga adalah *Naive Bayes Classification* (NBC) dengan dua varian algoritma yaitu varian *MultinomialNB* dan varian *Bernoulli*. Berdasarkan penelitian sebelumnya [31] dua varian model ini digunakan untuk meningkatkan nilai akurasi Naive Bayes pada klasifikasi sentimen dari 0.90% menjadi 0.92%. Berikut ini fungsi untuk menjalankan varian *MultinomialNB* dapat dilihat pada tabel 3.6.

Tabel 3. 6 MultinomialNB

Fungsi	Keterangan
from sklearn.naive_bayes *import MultinomialNB* *from sklearn import naive_bayes* *nb1 = naive_bayes.MultinomialNB()* *nb1.fit(X_train, y_train)* *y_pred_1 = nb1.predict(X_test)* *score_1 = metrics.accuracy_score* *(y_test, y_pred_1)* *print("Akurasi dengan menggunakan* *NB Multinomial: ", score_1)*	*Naive Bayes Classifier* *Varian MultinomialNB* Proses training data latih, prediksi data uji, perhitungan akurasi

Dan untuk *Naive Bayes Classification* (NBC) varian Bernoulli dapat dilihat pada tabel 3.7

Tabel 3. 7 Bernoulli

Fungsi	Keterangan
from sklearn.naive_bayes *import Bernoulli* *from sklearn import naive_bayes* *nb2 = naive_bayes.BernoulliNB* *nb2.fit(X_train, y_train)* *y_pred_2 = nb2.predict(X_test)* *score_2 = metrics.accuracy_score* *y_test, y_pred_2)* *print("Akurasi dengan menggunakan* *NB Bernoulli: ", score_2)*	*Naive Bayes Classifier* *Varian Bernoulli* Proses training data latih, prediksi data uji, perhitungan akurasi

3.6 Evaluasi Model

Evaluasi terhadap ketiga model : Support Vector Machine, Naive Bayes Classification, dan K-Nearest Neighbor menggunakan Classification Report, Confusion Matrix, Score Model Accuracy, dan Cross Validation Report. Classification report dan confusion matrix untuk menampilkan nilai Precision, Recall, F1-Score, Support, dan Accuracy. Dan evaluasi pada nilai akurasi pada setiap model akan menggunakan scor model accuracy.

a. Confusion Matrix

Confusion matrix sangat berguna untuk mengevaluasi pengklasifikasian. Kolom mewakili prediksi dan baris mewakili kelas yang sebenarnya [32]. Confusion matrix merupakan tabulasi data yang menggambarkan hasil klasifikasi pada suatu model[29]. Empat istilah yang merupakan representasi hasil proses klasifikasi pada confusion matrix yaitu *True Positive (TP), False Positive (FP), True Negative (TN),* dan *False Negative (FN)* seperti pada tabel 3.8 di bawah ini.

Tabel 3. 8 Confusion Matrix

Fungsi	Keterangan
True Positive (TP)	Nilai sesungguhnya adalah positif dan diprediksi positif
False Positive (FP)	Nilai sesungguhnya adalah negatif namun diprediksi positif
True Negative (TN)	Nilai sesungguhnya adalah negatif dan diprediksi negatif
False Negative (FN)	Nilai sesungguhnya adalah positif namun di prediksi negatif

Pada penelitian ini klasifikasi sentimen terdiri dari tiga kelas yaitu sentimen positif, negatif, dan netral seperti pada tabel 3.9 di bawah ini.

Tabel 3. 9 Confusion Matrix dengan Tiga Sentimen

Aktual	Prediksi		
	Negatif	Netral	Positif
Negatif	True Negatif	False Netral (1)	False Positif (1)
Netral	False Negatif (1)	True Netral	False Positif (2)
Positif	False Negatif (2)	False Netral (2)	True Positif

Berikut ini adalah fungsi yang digunakan untuk menampilkan *confusion matrix* untuk setiap model yang digunakan.

```
confusion_matric = confusion_matrix(y_test, y_pred)
print(confusion_matric)
```

1. Recall

Recall atau *sensitivity* merupakan rasio dari jumlah data positif yang diprediksi dengan benar (TP) terhadap jumlah data aktual yang bernilai positif atau dapat diformulasikan pada persamaan 3.6, 3.7, 3.8.

$$Recall\ .Positif = \frac{TP}{TP + FN1 + FNT}\ .\ (100\%) \tag{3.6}$$

$$Recall.Negatif = \frac{TN}{FP1 + TN + FNT}\ .\ (100\%) \tag{3.7}$$

$$Recall.Netral = \frac{TNT}{FP2 + FN2 + TNT}\ .\ (100\%) \tag{3.8}$$

2. Precision

Precision adalah rasio dari data positif yang diklasifikasikan bernilai benar terhadap jumlah total hasil prediksi positif seperti yang digambarkan pada persamaan 3.9,3.10,311.

$$Precision.Positif = \frac{TP}{TP + FP1 + FP2}\ .(100\%) \tag{3.9}$$

$$Precision.Negatif = \frac{TN}{FN1 + TN + FN2}\ .(100\%) \tag{3.10}$$

$$Precision.Netral = \frac{TNT}{FNT1 + FNT2 + TN} \ .(100\%) \qquad (3.11)$$

3. Specificity

Specificity merupakan rasio data yang diprediksi negatif dengan benar terhadap data aktual yang negatif atau dapat dikatakan sebagai *True Negative Rate* seperti persamaan 3.12.

$$Specificity = \frac{TP}{TN + TNT + FP} \ .(100\%) \qquad (3.12)$$

4. F1-score

F1-score menghitung hubungan antara *Precision* dan *Recall*, seperti yang dinyatakan pada persamaan 3.13.

$$F1 = 2\,x\,\frac{Precision\ x\ Recall}{Precision + Recall} \qquad (3.13)$$

5. Accuracy

Akurasi merupakan salah satu pengukuran model yang sangat penting dilakukan di dalam proses evaluasi model. Formulasi mencari nilai *accuracy* seperti pada persamaan 3.14.

$$Accuracy = \frac{TP + TN + TNT}{TP + FP + FN \ldots + TNT} \ .(100\%) \qquad (3.14)$$

6. Support

Support merupakan kemunculan aktual kelas didalam sebuah kumpulan data yang telah ditentukan. *Support* tidak merubah model melainkan mendiagnosa proses evaluasi

b. Score Model Accuracy

Score Model Accuracy yaitu untuk menampilkan nilai akurasi dari setiap model yang digunakan. Tabel 3.10 merupakan fungsi yang digunakan untuk menampilkan nilai akurasi masing-masing model.

Tabel 3. 10 Score Model Accuracy

Fungsi	Keterangan
clf = LinearSVC()	Score Model Accuracy untuk Model
clf.fit(X_train,y_train)	SVM
clfnb1 = MultinomialNB()	Score Model Accuracy untuk Model
clfnb1.fit(X_train, y_train)	Naive Bayes MultinomialNB
clfnb2 = BernoulliNB()	Score Model Accuracy untuk Model
clfnb2.fit(X_train,y_train)	Naive Bayes Bernoulli
clfknn = KNeighborsClassifier()	Score Model Accuracy untuk Model
clfknn.fit(X_train,y_train)	KNN

3.7 Cross Validation

Cross validation adalah metode statistik untuk mengevaluasi dan membandingkan algoritma *machine learning* dengan membagi data menjadi dua segmen: segmen pertama digunakan untuk mempelajari atau melatih model dan segmen kedua digunakan untuk memvalidasi model. Bentuk dasar dari *cross validation* adalah *k-fold* yaitu validasi terhadap *k-fold.* Pada *data mining* dan *machine learning*, *10 fold cross validation* adalah yang paling umum digunakan dan memiliki kinerja yang akurat [33]. Skema iterasi 10 fold validation dapat dilihat pada tabel 3.1 dibawah ini.

Tabel 3. 11 Skema 10 fold-validation

Seluruh Dataset									
Data Pengujian					Data Latih				
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

Dengan menggunakan library *python sklearn* berikut ini adalah *scripts program* yang dijalankan untuk menampilkan nilai score menggunakan *cross validation.*

```
def evaluate_model(X_train, y_test, repeats):
  cv = RepeatedKFold(n_splits=5, n_repeats=repeats, random_state
=1)
  model = svm.SVC()
  scores = cross_val_score(model, X_train, y_test, scoring='accu
racy', cv=cv, n_jobs=-1)
  return scores

X_train, y_test = make_classification()
repeats = range(1, 11)
results = list()
for r in repeats:
  scores = evaluate_model(X_train, y_test, r)
  print('>%d mean=%.4f se=%.3f' % (r, mean(scores), sem(scores))
)
  results.append(scores)
```

3.8 Word Cloud

Dalam proses ini setiap kata-kata yang paling banyak muncul di dalam sentimen positif, sentimen negatif, dan sentimen netral akan divisualisasikan dengan menggunakan *word clouds*. *Word clouds* menggambarkan banyaknya kata-kata yang muncul dan digambarkan dengan berbagai ukuran. Semakin besar ukuran kata-kata tersebut maka kata tersebut penting. *Library Python* yang digunakan untuk visualisasi *word clouds* adalah *matplotlib*. Berikut ini adalah *scripts program* yang digunakan untuk visualisasi.

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
```