



# **Analisis Sentimen Terhadap Tayangan Televisi Berdasarkan Opini Masyarakat pada Media Sosial Twitter menggunakan Algoritma Machine Learning Multinomial Naive Bayes**

---

Natural Language Processing

---

# Agenda

01

Latar Belakang Masalah, Tujuan,  
dan Urgensi

---

02

Dataset yang Digunakan

---

03

Preprocessing Data

---

04

Ekstraksi Fitur

---

05

Model yang Digunakan

---

06

Performa Model yang Dihasilkan


---

07

Kesimpulan

01

# Latar Belakang Masalah, Tujuan, Urgensi



---

---

# Latar Belakang Masalah

Televisi merupakan salah satu media elektronik yang menjadi hiburan dan sumber informasi melalui acara televisi yang ditayangkan. Dalam tayangan televisi tersebut terdapat beberapa penilaian berdasarkan opini yang diberikan oleh masyarakat melalui media sosial Twitter. Penilaian tersebut memiliki sentiment yang positive maupun negative. Oleh karena itu, dalam ujian praktik ini melakukan klasifikasi teks dengan menggunakan algoritma MultinomialNB yang diharapkan dapat menunjukkan hasil persentase akurasi yang baik dan memprediksikan sentiment tayangan televisi yang bernilai negative atau positive berdasarkan opini masyarakat pada media sosial Twitter.

---

## Tujuan & Urgensi

Tujuan dari latar belakang masalah tersebut untuk melakukan klasifikasi teks dengan menggunakan algoritma MultinomialNB untuk memprediksikan sentiment opini masyarakat mengenai tayangan televisi yang bernilai negative atau positive.

02

# Dataset yang Digunakan



# Dataset yang digunakan.




rizalespe / Dataset-Sentimen-Analisis-Bahasa-IndonesiaPublic

NotificationsFork 70Star 41

<> CodeIssuesPull requestsActionsProjectsSecurityInsights

masterDataset-Sentimen-Analisis-Bahasa-Indonesia / dataset\_tweet\_sentimen\_tayangan\_tv.csvGo to file

 rizalespe Add files via upload ✓

Latest commit 413b888 on Jan 4, 2020History

1 contributor

401 lines (401 sloc)45.7 KBRawBlame

Search this file...

	Id	Sentiment	Acara TV	Jumlah Retweet	Text Tweet
1	1	positive	HitamPutihTransTV	12	Undang @N_ShaniJKT48 ke hitamputih, pemenang SSK JKT48 harusnya mJKT48 ini lebih Layak di Undang karena prestasinya
2	2	positive	HitamPutihTransTV	6	Selamat berbuka puasa Semoga amal ibadah hari ni diterima Allah #hitamputih
3	3	positive	HitamPutihTransTV	9	Ada nih di trans7 hitam putih, dia dpt penghargaan juga di norwegia #hitamputih
4	4	positive	HitamPutihTransTV	2	selamat ya mas @adietaufan masuk hitamputih
5	5	positive	HitamPutihTransTV	1	Asiknya nonton Hitam Putih Trans7

Dataset yang digunakan adalah **dataset\_tweet\_sentimen\_tayangan\_tv.csv** yang terdiri dari 400 data jumlah sentiment acara TV dengan sentiment negative : 200 data dan sentiment positive : 200 data.

03

# Preprocessing Data

---



---

- Case Folding
- Filtering
- Stopword
- Stemming

# Case Folding

```
import re

# Buat fungsi untuk langkah case folding
def casefolding(text):
    text = text.lower() # Mengubah teks menjadi lower case
    text = re.sub(r'https?:\/\/\S+|www\.\S+', '', text) # Menghapus URL
    text = re.sub(r'[-+]?[0-9]+', '', text) # Menghapus angka
    text = re.sub(r'^\w\s', '', text) # Menghapus karakter tanda baca
    text = text.strip()
    return text

raw_sample = data['Text Tweet'].iloc[80]
case_folding = casefolding(raw_sample)

print('Raw data\t: ', raw_sample)
print('\nCase folding\t: ', case_folding)

Raw data      :  toleransi umat beragama macam apa yg diterapkan di Serang Banten? Toleransi hanya mitos #hitamPutihT7
Case folding   :  toleransi umat beragama macam apa yg diterapkan di serang banten toleransi hanya mitos hitamputiht
```

Melakukan Case Folding untuk memproses text preprocessing dengan mengubah karakter pada data, yaitu :

- 1. Mengubah teks menjadi lower case
- 2. Menghapus URL pada text
- 3. Menghapus angka
- 4. Menghapus karakter tanda baca

# Word Normalization

```
# Download corpus kumpulan slangwords
!wget https://raw.githubusercontent.com/ksnugroho/klasifikasi-spam-sms/master/data/key_norm.csv

--2022-10-05 13:22:37-- https://raw.githubusercontent.com/ksnugroho/klasifikasi-spam-sms/master/data/key_norm.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 80969 (79K) [text/plain]
Saving to: 'key_norm.csv'

key_norm.csv      100%[=====>] 79.07K  --.-KB/s    in 0.01s

2022-10-05 13:22:37 (5.65 MB/s) - 'key_norm.csv' saved [80969/80969]

# Mengubah kata tidak baku menjadi kata baku
key_norm = pd.read_csv('https://raw.githubusercontent.com/ksnugroho/klasifikasi-spam-sms/master/data/key_norm.csv')
print(key_norm.head(10))

key_norm.shape

   _id  singkat  hasil
0    1    abis    habis
1    2  accent  tekanan
2    3  accept   terima
3    4  accident kecelakaan
4    5 achievement prestasi
5    6    acra    acara
```

Word normalization berfungsi untuk mengubah kata tidak baku menjadi kata baku dari kumpulan slangwords yang sudah tersedia.



# Filtering (Stopword Removal)

```
# Import library yang akan digunakan untuk melakukan stopwords removal
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords

stopwords_ind = stopwords.words('indonesian')

# Menampilkan panjang jumlah stopwords indonesia yang disediakan NLTK
len(stopwords_ind)

758

# Lihat daftar stopwords yang disediakan NLTK
stopwords_ind

['ada',
 'adalah',
 'adanya',
 'adapun',
 'agak',
 'agakny',
 'agar',
 'akan',
 'akankah',
 'akhir',
 'akhiri',
```

```
# Menambahkan kata lain dalam stopwords yang ingin dihilangkan
more_stopword = ['apa', 'banget', 'bego', 'cuma', 'cukup', 'dg', 'deh', 'bkin', 'usak', 'kok', 'klo', 'ga', 'gpp', 'gapapa', 'smoga']
stopwords_ind = stopwords_ind + more_stopword

# Buat fungsi untuk langkah stopwords removal
def remove_stop_words(text):
    clean_words = []
    text = text.split()
    for word in text:
        if word not in stopwords_ind:
            clean_words.append(word)
    return " ".join(clean_words)

raw_sample = data['Text Tweet'].iloc[80]
case_folding = casefolding(raw_sample)
stopword_removal = remove_stop_words(case_folding)

print('Raw data\t\t: ', raw_sample)
print('\nCase folding\t\t: ', case_folding)
print('\nStopword removal\t: ', stopword_removal)

Raw data          : toleransi umat beragama macam apa yg diterapkan di Serang Banten? Toleransi hanya mitos #hitamPutihT7

Case folding      : toleransi umat beragama macam apa yg diterapkan di serang banten toleransi hanya mitos hitamputiht

Stopword removal  : toleransi umat beragama diterapkan serang banten toleransi mitos hitamputiht
```

Filtering berfungsi untuk menghilangkan kata-kata yang tidak penting dalam Text Tweet dengan menambahkan kata lain dalam stopwords

# Text Preprocessing Pipeline

```
def text_preprocessing_process(text):
    text = casefolding(text)
    text = text_normalize(text)
    text = remove_stop_words(text)
    text = stemming(text)
    return text
```

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

```
# Buat fungsi untuk langkah stemming bahasa Indonesia
def stemming(text):
    text = stemmer.stem(text)
    return text
```

```
data['clean_teks'] = data['Text Tweet'].apply(text_preprocessing_process)
```

```
# Perhatikan waktu komputasi ketika proses text preprocessing
```

```
raw_sample = data['Text Tweet'].iloc[80]
case_folding = casefolding(raw_sample)
stopword_removal = remove_stop_words(case_folding)
text_stemming = stemming(stopword_removal)
```

```
CPU times: user 1min 16s, sys: 245 ms, total: 1min 16s
Wall time: 1min 18s
```

```
print('Raw data\t\t: ', raw_sample)
print('Case folding\t\t: ', case_folding)
print('Stopword removal\t: ', stopwords_removal)
print('Stemming\t\t: ', text_stemming)
```

```
# Menampilkan dataset
data
```

```
Raw data      : toleransi umat beragama macam apa yg diterapkan di Serang Banten? Toleransi hanya mitos #hitamPutihT7
Case folding  : toleransi umat beragama macam apa yg diterapkan di serang banten toleransi hanya mitos hitamputiht
Stopword removal : toleransi umat beragama diterapkan serang banten toleransi mitos hitamputiht
Stemming      : toleransi umat agama terap serang banten toleransi mitos hitamputiht
```

	Id	Sentiment	Acara TV	Jumlah Retweet	Text Tweet	clean_teks
	0	1	HitamPutihTransTV	12	Undang @N_ShaniJKT48 ke hitamputih, pemenang S...	undang n shanijkt hitamputih menang ssk jakart...
	1	2	HitamPutihTransTV	6	Selamat berbuka puasa Semoga amal ibadah hari ...	selamat buka puasa moga amal ibadah terima all...
	2	3	HitamPutihTransTV	9	Ada nih di trans7 hitam putih, dia dpt penghar...	trans hitam putih harga norwegia hitamputih
	3	4	HitamPutihTransTV	2	selamat ya mas @adietaufan masuk hitamputih	selamat mas adietaufan masuk hitamputih
	4	5	HitamPutihTransTV	1	Asiknya nonton Hitam Putih Trans7	asiknya nonton hitam putih trans
	...	...	...	...	...	...
	395	396	0 MataNajwaMetroTV	0	ini apa banget deh gw paling kesel klo orang2 ...	kesel orang debat pakai emosi matanajwametrotv
	396	397	0 MataNajwaMetroTV	0	Orang miskin semakin miskin klo sekolah melaku...	orang miskin miskin sekolah pungut liar
	397	398	0 MataNajwaMetroTV	0	ga boLeh emosi, cepat tua, nonton #matanajwame...	emosi cepat tua nonton matanajwametrotv lihat ...
	398	399	0 MataNajwaMetroTV	0	dr penampilan saja kyk preman taunya bkin kistr...	tampil preman tau kistruh usak matanajwametrotv
	399	400	0 MataNajwaMetroTV	0	Jawab aja ga usah berbelit-belit. Muter2 ga je...	berbelitbelit putar buang mutu matanajwametrotv

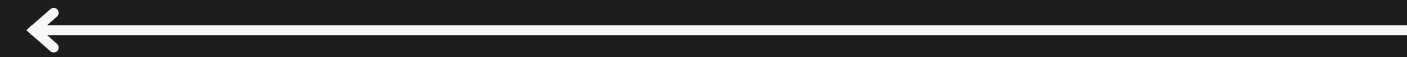
400 rows x 8 columns

```
# Menyimpan data yang telah melalui text preprocessing agar kita tidak perlu menjalankan proses tersebut mulai awal (Opsional)
data.to_csv('clean_data.csv')
```

Stemming berfungsi untuk mengubah kata menjadi kata dasarnya

# Ekstraksi Fitur

---



- Feature Extraction – BoW & TF IDF
- Feature Selection – Chi-Square

# Feature Engineering

Memisahkan kolom fitur dan target dengan variabel X menampilkan 'clean\_teks' dan variabel y menampilkan 'Sentiment' yang sudah diberikan label berupa numerik dengan menggunakan library Label Encoder yang memiliki keterangan 1 adalah sentiment positive dan 0 adalah sentiment negative.

```
# Pisahkan kolom fitur dan target
X = data['clean_teks']
y = data['Sentiment']
```

X

```
0      undang n shanijkt hitamputih menang ssk jakart...
1      selamat buka puasa moga amal ibadah terima all...
2      trans hitam putih harga norwegia hitamputih
3      selamat mas adietaufan masuk hitamputih
4      asiknya nonton hitam putih trans
...
395     kesel orang debat pakai emosi matanajwametrotv
396     orang miskin miskin sekolah pungut liar
397     emosi cepat tua nonton matanajwametrotv lihat ...
398     tampil preman tau kisruh usak matanajwametrotv
399     berbelitbelit putar buang mutu matanajwametrotv
Name: clean_teks, Length: 400, dtype: object
```

y

```
0      1
1      1
2      1
3      1
4      1
...
395     0
396     0
397     0
398     0
399     0
Name: Sentiment, Length: 400, dtype: int64
```

# Feature Extraction

Feature Extraction (Bag of Words & N-Gram) berfungsi untuk memproses teks menjadi vektor menggunakan metode BoW dengan 'ngram\_range=(1,2)' yang menghasilkan 3368 jumlah fitur. Kemudian, melihat fitur-fitur yang tersedia dalam corpus dan matriks jumlah fitur. Data tersebut akan dimasukkan dalam proses pemodelan Machine Learning.

```
...
Convert a collection of text documents to a matrix of token counts.
https://scikit-learn.org/stable/modules/generated/sklearn.feature\_extraction.text.CountVectorizer.html
'''
from sklearn.feature_extraction.text import CountVectorizer

# BoW - bigram
bow = CountVectorizer(ngram_range=(1,2))
bow.fit(X)

CountVectorizer(ngram_range=(1, 2))

# Melihat jumlah fitur
print(len(bow.get_feature_names_out()))

3368

# Melihat fitur-fitur apa saja yang ada di dalam corpus
print(bow.get_feature_names_out())

['aa' 'aa gym' 'aa gymnastiar' ... 'zaitun rasmin' 'zhonk'
'zhonk kickandymetrotv']
```

```
X_bow = bow.transform(X).toarray()
X_bow
```

```
array([[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, 0],
       [0, 0, 0, ..., 0, 0, 0]])
```

```
data_bow = pd.DataFrame(X_bow, columns=bow.get_feature_names_out())
data_bow
```

	aa	aa gym	aa gymnastiar	aagym	aagym partai	abang	abang acara	abas	abas anjay	abi	...	yuk
0	0	0	0	0	0	0	0	0	0	0	...	0
1	0	0	0	0	0	0	0	0	0	0	...	0
2	0	0	0	0	0	0	0	0	0	0	...	0
3	0	0	0	0	0	0	0	0	0	0	...	0
4	0	0	0	0	0	0	0	0	0	0	...	0

```
with open('bow_feature.pickle', 'wb') as output:
    pickle.dump(X_bow, output)
```



# Feature Extraction

Feature Extraction (TF-IDF & N-Gram) berfungsi untuk mengubah teks menjadi vector menggunakan metode TF-IDF dengan 'ngram\_range=(1,2)' yang menghasilkan 3368 jumlah fitur. Kemudian, melihat fitur-fitur yang tersedia dalam corpus dan matriks jumlah token. Data tersebut akan dimasukkan dalam proses pemodelan Machine Learning.

```
'''
Convert a collection of raw documents to a matrix of TF-IDF features
https://scikit-learn.org/stable/modules/generated/sklearn.feature\_extraction.text.TfidfVectorizer.html
'''

from sklearn.feature_extraction.text import TfidfVectorizer

tf_idf = TfidfVectorizer(ngram_range=(1,2))
tf_idf.fit(X)

TfidfVectorizer(ngram_range=(1, 2))

# Melihat Jumlah Fitur
print(len(tf_idf.get_feature_names_out()))

3368

# Melihat fitur-fitur apa saja yang ada di dalam corpus
print(tf_idf.get_feature_names_out())

['aa' 'aa gym' 'aa gymnastiar' ... 'zaitun rasmin' 'zhonk'
'zhonk kickandymetrotv']
```

```
X_tf_idf = tf_idf.transform(X).toarray()
X_tf_idf

array([[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., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])

# Melihat matriks jumlah token menggunakan TF IDF, lihat perbedaannya dengan metode Bow
# Data ini siap untuk dimasukkan dalam proses pemodelan (machine learning)

data_tf_idf = pd.DataFrame(X_tf_idf, columns=tf_idf.get_feature_names_out())
data_tf_idf
```

	aa	aa gym	aa gymnastiar	aagym	aagym partai	abang	abang acara	abas	abas anjay	abi
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
2	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
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

```
with open('tf_idf_feature.pickle', 'wb') as output:
    pickle.dump(X_tf_idf, output)
```

# Feature Selection (Chi-Square)

```
# Mengubah nilai data tabular tf-idf menjadi array agar dapat dijalankan pada proses seleksi fitur
X = np.array(data_tf_idf)
y = np.array(y)

...

Select features according to the k highest scores.
https://scikit-learn.org/stable/modules/generated/sklearn.feature\_selection.SelectKBest.html

Compute chi-squared stats between each non-negative feature and class.
https://scikit-learn.org/stable/modules/generated/sklearn.feature\_selection.chi2.html
...

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

# Ten features with highest chi-squared statistics are selected
chi2_features = SelectKBest(chi2, k=500)
X_kbest_features = chi2_features.fit_transform(X, y)

# Reduced features
print('Original feature number:', X.shape[1])
print('Reduced feature number:', X_kbest_features.shape[1])
```

Output :

```
Original feature number: 3368
Reduced feature number: 500
```

Feature Selection (Chi-Square) berfungsi untuk menyeleksi fitur dengan menggunakan fungsi `x_kbest_features` dimana `k` yang digunakan adalah 500 sehingga dari 3368 fitur menjadi 500 fitur.

# Feature Selection (Chi-Square)

```
# chi2_features.scores_ adalah nilai chi-square, semakin tinggi nilainya maka semakin baik fiturnya
data_chi2 = pd.DataFrame(chi2_features.scores_, columns=['nilai'])
data_chi2
```

	nilai
0	0.793109
1	0.521906
2	0.333342
3	0.244436
4	0.244436
...	...
3363	0.330585
3364	0.351302

```
# Menampilkan fitur beserta nilainya
feature = tf_idf.get_feature_names_out()
data_chi2['fitur'] = feature
data_chi2
```

	nilai	fitur
0	0.793109	aa
1	0.521906	aa gym
2	0.333342	aa gymnastiar
3	0.244436	aagym
4	0.244436	aagym partai
...	...	...
3363	0.330585	yukikatou nongol
3364	0.351302	zaitun
3365	0.351302	zaitun rasmin
3366	0.229672	zhonk
3367	0.229672	zhonk kickandymetrotv

3368 rows × 2 columns

```
# Mengurutkan fitur terbaik
data_chi2.sort_values(by='nilai', ascending=False)
```

	nilai	fitur
1461	5.706543e+00	keren
1048	5.354965e+00	hitamputiht
1187	4.262589e+00	inspirasi
1536	3.956864e+00	kickandymetrotv
1039	3.652803e+00	hitam putih
...	...	...
828	8.823291e-05	episode
2388	1.409085e-05	pihak
1370	1.268364e-05	kandang
2832	6.887770e-06	si
3038	6.284208e-10	televisi acara

Pada Chi-Square mengurutkan nilai yang tertinggi. Semakin tinggi nilainya maka semakin baik fiturnya. Selanjutnya menampilkan fitur beserta nilainya dan mengurutkan



# Feature Selection (Chi-Square)

```
# Menampilkan fitur-fitur yang sudah diseleksi
# Beserta nilai vektornya pada keseluruhan data untuk dijalankan pada proses machine learning

# Hanya k fitur yang terpilih sesuai parameter k yang ditentukan sebelumnya

data_selected_feature = pd.DataFrame(X_kbest_features, columns=selected_feature)
data_selected_feature
```

	aa	aa gym	abraham	acara	acara hitam	acara inspirasi	acara televisi	adem	indonesialawyersclub	adem adil	adil
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
2	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
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0
...	...	...	...	...	...	...	...	...		...	...
395	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0
396	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0
397	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0
398	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0
399	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0		0.0	0.0

Pada output diatas merupakan fitur-fitur yang sudah diseleksi beserta nilai vektornya pada keseluruhan data dari DataFrame 'X\_kbest\_features'.

05

# Model yang Digunakan



# Model yang Digunakan

```
# Training the model
algorithm = MultinomialNB()
model = algorithm.fit(X_train, y_train)

# Load algoritma pembelajaran
# Fitkan (latih) algoritma pada data latih & label latih

# Simpan model hasil traning
dump(model, filename='model_1.joblib')

['model_1.joblib']

# Gunakan model yang telah di latih untuk memprediksi label pada data uji
model_pred = model.predict(X_test)

# Tampilkan hasil prediksi label dari model
model_pred

array([0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1])

# Tampilkan label sebenarnya pada data uji (actual label)
y_test

array([0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1])
```

Melakukan training model dengan menggunakan algoritma MultinomialNB dengan random\_state sebesar 70 dan test\_size sebesar 0.2

# Evaluasi Model

Pada evaluasi model dengan menggunakan algoritma MultinomialNB diperoleh akurasi pengujian yang dihasilkan dari algoritma multinomial naive bayes sebesar 83.75% dengan jumlah prediksi benar 67 dan jumlah prediksi salah 13

```
# Hitung jumlah data yang berhasil di prediksi model & jumlah data yang salah di prediksi
prediksi_benar = (model_pred == y_test).sum()
prediksi_salah = (model_pred != y_test).sum()
```

```
print('Jumlah prediksi benar\t:', prediksi_benar)
print('Jumlah prediksi salah\t:', prediksi_salah)
```

```
accuracy = prediksi_benar / (prediksi_benar + prediksi_salah)*100
print('Akurasi pengujian\t:', accuracy, '%')
```

```
Jumlah prediksi benar    : 67
Jumlah prediksi salah    : 13
Akurasi pengujian        : 83.75 %
```

```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, model_pred)
print('Confusion matrix:\n', cm)
```

```
Confusion matrix:
[[31  6]
 [ 7 36]]
```

```
from sklearn.metrics import classification_report
```

```
print('Classification report:\n', classification_report(y_test, model_pred))
```

```
Classification report:
              precision    recall  f1-score   support

     0           0.82       0.84       0.83         37
     1           0.86       0.84       0.85         43

   accuracy                   0.84         80
  macro avg           0.84       0.84       0.84         80
 weighted avg           0.84       0.84       0.84         80
```

06

# Performa Model yang Dihasilkan



# ●● Performa Model yang Dihasilkan

```
# Cross Validation

from sklearn.model_selection import ShuffleSplit    # bisa pilih beberapa teknik cross validation
from sklearn.model_selection import cross_val_score # untuk mengetahui performa model pada cross validation

cv = ShuffleSplit(n_splits=10, test_size=0.2, random_state=70)

cv_accuracy = (cross_val_score(model, X_kbest_features, y, cv=cv, scoring='accuracy'))
avg_accuracy = np.mean(cv_accuracy)

print('Akurasi setiap split:', cv_accuracy, '\n')
print('Rata-rata akurasi pada cross validation:', avg_accuracy)
```

```
Akurasi setiap split: [0.8375 0.8625 0.9375 0.8      0.9      0.9      0.9125 0.9      0.875  0.875 ]
```

```
Rata-rata akurasi pada cross validation: 0.8800000000000001
```

Pada output diatas menghasilkan akurasi setiap split dan rata-rata akurasi pada Cross Validation, yaitu 0.880

# ●● Performa Model yang Dihasilkan

Masukkan Teks Anda:

```
[▶] #@title Masukkan Teks Anda:
# Prediksi Sentimen Negative

input_text = input("Masukkan Text Tweet : ")

pre_input_text = text_preprocessing_process(input_text) # lakukan text pre processing pada text input

tf_idf_vec = TfidfVectorizer(vocabulary=set(vocab))      # definisikan TF_IDF

result = model.predict(tf_idf_vec.fit_transform([pre_input_text])) # Lakukan prediksi

print('\nHasil Text Preprocessing :', pre_input_text)

if (result==0):
    sentimen = 'negative'
else:
    sentimen = 'positive'

print('\nHasil prediksi: ', input_text, ' adalah ', sentimen)
```

```
↳ Masukkan Text Tweet : kesel orang debat pakai emosi matanajwametrotv

Hasil Text Preprocessing : kesel orang debat pakai emosi matanajwametrotv

Hasil prediksi:  kesel orang debat pakai emosi matanajwametrotv  adalah  negative
```

Masukkan Teks Anda:

```
[191] #@title Masukkan Teks Anda:
# Prediksi Sentimen Positive

input_text = input("Masukkan Text Tweet : ")

pre_input_text = text_preprocessing_process(input_text) # lakukan text pre processing pada text input

tf_idf_vec = TfidfVectorizer(vocabulary=set(vocab))      # definisikan TF_IDF

result = model.predict(tf_idf_vec.fit_transform([pre_input_text])) # Lakukan prediksi

print('\nHasil Text Preprocessing :', pre_input_text)

if (result==0):
    sentimen = 'negative'
else:
    sentimen = 'positive'

print('\nHasil prediksi: ', input_text, ' adalah ', sentimen)
```

```
Masukkan Text Tweet : keren undang acara hitamputih acara presiden news trans

Hasil Text Preprocessing : keren undang acara hitamputih acara presiden berita trans

Hasil prediksi:  keren undang acara hitamputih acara presiden news trans  adalah  positive
```

Pada code diatas melakukan prediksi pada tahap Deployment dengan memasukkan Text Tweet yang telah dilakukan text preprocessing. Hasil prediksi dari Teks yang dimasukkan berupa sentiment Negative dan Positive.

---

07

# Kesimpulan

---



**Berdasarkan "Analisis Sentimen Terhadap Tayangan Televisi**

**Berdasarkan Opini Masyarakat pada Media Sosial Twitter menggunakan**

**Algoritma Machine Learning Multinomial Naive Bayes" yang telah**

**dilakukan diperoleh akurasi sebesar 0.84 sudah bagus untuk melakukan**

**prediksi sentiment Negative dan Positive pada Text Tweet dengan tepat**

**dimana jumlah prediksi benar yang dihasilkan adalah 67 dan jumlah**

**prediksi salah yang dihasilkan adalah 13.**



# Thank you!

---

---

