Sentiment Analysis pada Data Review Film

Oleh: EKO ZULKARYANTO

Pada kesempatan ini dilakukan uji kinerja atau akurasi pada beberapa algoritma pengenalan pola pada teks untuk data review film. Data diunduh dari https://drive.google.com/file/d/1lp9TXFEbA2Yy8vA8PDSsrarvprw0GIbQ/ view?usp=sharing. Data review sudah dikelompokkan ke dalam dua folder, yaitu baik dan jelek. Masing-masing folder berisi 500 data teks review film.

Proses pengolahan data menggunakan Jupyter Notebook versi offline atau dapat menggunakan versi online di https://jupyter.org/try.

Langkah pertama adalah membuat DataFrame, yaitu dengan membaca file-file data tersebut kemudian disimpan ke dalam list. Nama folder ditetapkan sebagai fitur *class* dan isi file sebagai fitur *text*.

```
import numpy
import pandas
import os

row_list = []
for subdir in ['bagus','jelek']:
    for folder, subfolders, filenames in os.walk(review_film/'+subdir):
        for file in filenames:
            d = {'class':subdir}
            with open('review_film/'+subdir+'/'+file) as f:
                if f.read():
                    f.seek(0)
                    d['text'] = f.read()
            row_list.append(d)
        break
```

Selanjutnya list tersebut dimasukkan ke dalam objek DataFrame.

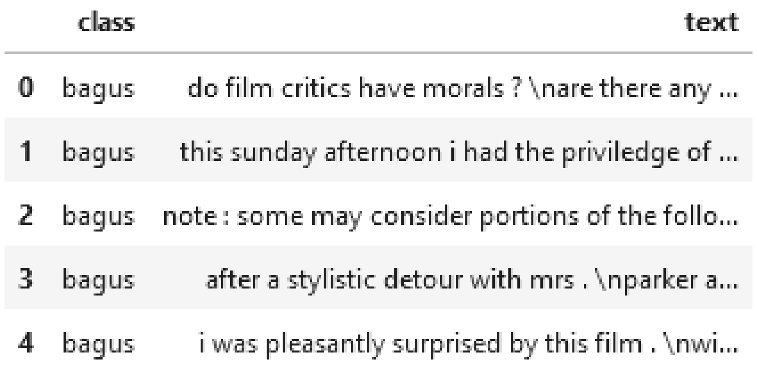
```
dataframe = pandas.DataFrame(row_list)
```

Untuk melihat banyaknya dataframe, dijalankan perintah `print(len(dataframe))`, sehingga diperoleh 1000.

Kemudian kita lihat isi DataFrame tersebut.

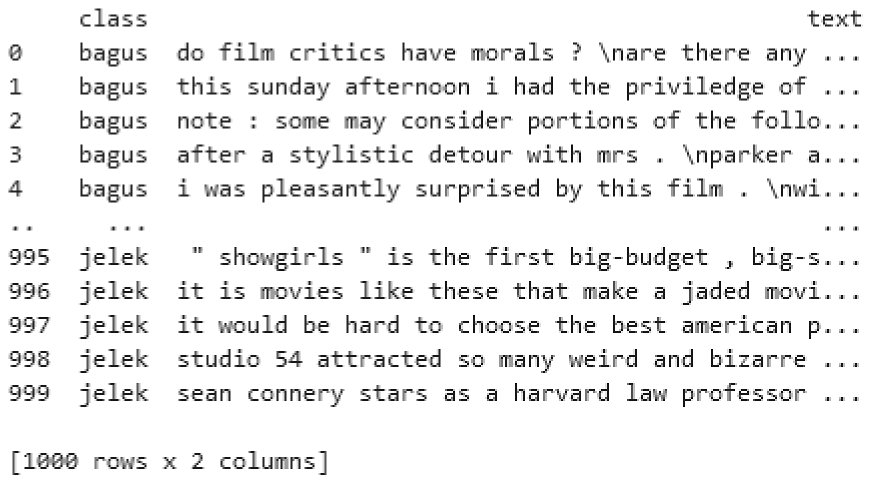
```
dataframe.head()
```

Ouputnya:



	class	text
0	bagus	do film critics have morals ? \nare there any ...
1	bagus	this sunday afternoon i had the priviledge of ...
2	bagus	note : some may consider portions of the follo...
3	bagus	after a stylistic detour with mrs . \nparker a...
4	bagus	i was pleasantly surprised by this film . \nwi...

Atau dengan perintah `print(dataframe)` dengan hasil seperti di bawah ini:



```
     class                                               text
0    bagus  do film critics have morals ? \nare there any ...
1    bagus  this sunday afternoon i had the priviledge of ...
2    bagus  note : some may consider portions of the follo...
3    bagus  after a stylistic detour with mrs . \nparker a...
4    bagus  i was pleasantly surprised by this film . \nwi...
..     ...                                                ...
995  jelek   " showgirls " is the first big-budget , big-s...
996  jelek  it is movies like these that make a jaded movi...
997  jelek  it would be hard to choose the best american p...
998  jelek  studio 54 attracted so many weird and bizarre ...
999  jelek  sean connery stars as a harvard law professor ...

[1000 rows x 2 columns]
```

Selanjutnya dilakukan Ekstrak Fitur (***Feature Extraction***) terhadap fitur *text*.

```
from sklearn.feature_extraction.text import CountVectorizer

count_vectorizer = CountVectorizer()
counts = count_vectorizer.fit_transform(dataframe['text'])
```

Kemudian dilakukan ***Training*** klasifikasi dengan **Multinomial Naïve Bayes** dengan target *class*.

```
from sklearn.naive_bayes import MultinomialNB

classifier = MultinomialNB()
targets = dataframe['class']
classifier.fit(counts, targets)
```

Output:

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Selanjutnya dilakukan ***testing*** terhadap data *testing*, pada bagian ini 2 contoh data *testing*.

```
examples = ["the law of crowd pleasing romantic movies states that the two
leads must end up togetherby film's end .if you're not familiar with this
law , then maybe you've seen the trailer for this film which shows that the
two leads are together by film's end . now if you're a regular reader of
mine , you've heard me say this countless times : you know how drive me
crazy is going to end , but is the journey to get to that ending worth it ?
no , it definitely is not . melissa joan hart ( from abc's  sabrina , the
teenage witch  ) likes a hunky stud on the basketball team . adrien grenier
is her grungy neighbor who's just broken up with his activist girlfriend .
apparently he wants to make his ex-girlfriend jealous enough to take him
back , and she wants someone to take her to the big year end dance .",
"this three hour movie opens up with a view of singer/guitar
player/musician/composer frank zappa rehearsing with his fellow band
members . all the rest displays a compilation of footage , mostly from the
concert at the palladium in new york city , halloween 1979 . other footage
shows backstage foolishness , and amazing clay animation by bruce bickford
. the performance of \" titties and beer \" played in this movie is very
entertaining , with drummer terry bozzio supplying the voice of the devil .
frank's guitar solos outdo any van halen or hendrix i've ever heard . bruce
bickford's outlandish clay animation is that beyond belief with zooms ,
morphings , etc . and actually , it doesn't even look like clay , it looks
like meat ."]

example_counts = count_vectorizer.transform(examples)

predictions = classifier.predict(example_counts)

print(predictions)
```

Output:

```
['jelek' 'bagus']
```

Selanjutnya dilakukan **PipeLine** terhadap data contoh di atas.

```
from sklearn.pipeline import Pipeline

pipeline = Pipeline([
    ('vectorizer',  CountVectorizer()),
    ('classifier',  MultinomialNB()) ])

pipeline.fit(dataframe['text'], dataframe['class'])
print(pipeline.predict(examples))
```

Output:

```
['jelek' 'bagus']
```

Selanjutnya kita dapat lakukan **validasi** untuk mengetahui kinerja dari algoritma klasifikasi Multinomial Naïve Bayes ini menggunakan Model Selection KFold.

```
from sklearn.model_selection import KFold
from sklearn.metrics import confusion_matrix, f1_score

k_fold = KFold(n_splits=6, random_state=None, shuffle=True)
scores = []
confusion = numpy.array([[0, 0], [0, 0]])
for train_indices, test_indices in k_fold.split(dataframe):
    train_text = dataframe.iloc[train_indices]['text']
    train_y = dataframe.iloc[train_indices]['class']

    test_text = dataframe.iloc[test_indices]['text']
    test_y = dataframe.iloc[test_indices]['class']

    pipeline.fit(train_text, train_y)
    predictions = pipeline.predict(test_text)

    confusion += confusion_matrix(test_y, predictions)
    score = f1_score(test_y, predictions, pos_label="jelek")
    scores.append(score)

print('Total review classified:', len(dataframe))
print('Score:', sum(scores)/len(scores))
print('Confusion matrix:')
print(confusion)
```

Berikut *output* hasil validasinya:

```
Total review classified: 1000
Score: 0.8129133784689189
Confusion matrix:
[[397 103]
 [ 79 421]]
```

Skor yang diperoleh adalah **81.29%**. Hasil ini sudah cukup bagus.

Selanjutnya kita juga dapat meningkatkan hasil akurasinya dengan **memperbaiki pipeline**. Diterapkan n-gram, 1 sampai 2, atau disebut juga bigram count. Yang berarti satu sampai dua kata yang akan di Vectorize.

```
pipeline = Pipeline([
    ('count_vectorizer', CountVectorizer(ngram_range=(1, 2))),
    ('classifier',       MultinomialNB())
])
```

Sehingga hasil validasinya menjadi:

```
Total review classified: 1000
Score: 0.8280299225317368
Confusion matrix:
[[428  72]
 [105 395]]
```

Diperoleh hasil skornya meningkat menjadi **82.80%.**

Kemudian dicoba juga **diterapkan TfidfTransformer**.

```
from sklearn.feature_extraction.text import TfidfTransformer

pipeline = Pipeline([
    ('count_vectorizer',   CountVectorizer(ngram_range=(1,  2))),
    ('tfidf_transformer',  TfidfTransformer()),
    ('classifier',         MultinomialNB())
])
```

Output:

```
Total review classified: 1000
Score: 0.7550366789373735
Confusion matrix:
[[365 135]
 [ 95 405]]
```

Berdasarkan hasil di atas, justru skor validasinya **menurun** menjadi 75.50%.

Selanjutnya dilakukan juga **menggunakan classifier Bernoulli Naïve Bayes.**

```
from sklearn.naive_bayes import BernoulliNB

pipeline = Pipeline([
    ('count_vectorizer',   CountVectorizer(ngram_range=(1, 2))),
    ('classifier',         BernoulliNB(binarize=0.0)) ])
```

Output:

```
Total review classified: 1000
Score: 0.6698751195307312
Confusion matrix:
[[263 237]
 [ 19 481]]
```

Berdasarkan hasil tersebut, skornya **66.98%.**

Kesimpulan

Berdasarkan beberapa percobaan untuk peningkatan kinerja di atas diperoleh seperti table di bawah ini:

Features	Classifier	False Jelek	False Bagus	F1 score
Bag of words counts	MultinomialNB	79	103	0.8129133784689189
Bigram counts	MultinomialNB	105	72	0.8280299225317368
Bigram frequencies	MultinomialNB	95	135	0.7550366789373735
Bigram occurrences	BernoulliNB	19	237	0.6698751195307312

Jadi, peningkatan dengan bigram counts dan classifier Multinomial Naïve bayes mendapatkan F1 Score yang tertinggi, yaitu **82.80%.**

MEMPREDIKSI CONTOH REVIEW

Untuk memprediksi review, yaitu dengan kode di bawah ini:

```
tes = ["the law of crowd pleasing romantic movies states that the two
leads must end up togetherby film's end .if you're not familiar with
this law , then maybe you've seen the trailer for this film which
shows that the two leads are together by film's end . now if you're a
regular reader of mine , you've heard me say this countless times :
you know how drive me crazy is going to end , but is the journey to
get to that ending worth it ? no , it definitely is not . melissa joan
hart ( from abc's  sabrina , the teenage witch  ) likes a hunky stud
on the basketball team . adrien grenier is her grungy neighbor who's
just broken up with his activist girlfriend . apparently he wants to
make his ex-girlfriend jealous enough to take him back , and she wants
someone to take her to the big year end dance ."]
tes_counts = count_vectorizer.transform(tes)
predictions = classifier.predict(tes_counts)
print(predictions)
```

Output:

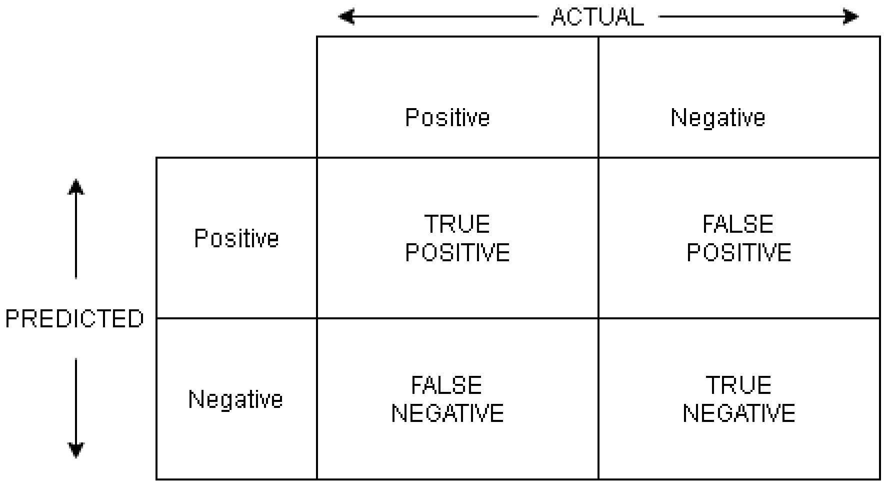
```
['jelek']
```

Jadi, review tersebut masuk dalam *class* jelek.

PENJELASAN

Confusion Matrix adalah suatu metode pengukuran performa untuk masalah klasifikasi *machine learning* dimana keluaran dapat berupa dua kelas atau lebih, dan berupa tabel dengan 4 kombinasi berbeda dari nilai prediksi dan nilai aktual.

Ada empat istilah yang merupakan representasi hasil proses klasifikasi pada confusion matrix yaitu True Positif, True Negatif, False Positif, dan False Negatif.



		ACTUAL	
		Positive	Negative
PREDICTED	Positive	TRUE POSITIVE	FALSE POSITIVE
	Negative	FALSE NEGATIVE	TRUE NEGATIVE

Sehingga, untuk menghitung presisi menggunakan rumus:

$$\text{PRECISION} = \frac{\begin{array}{c}\text{TRUE POSITIVES}\\ \text{(TP)}\end{array}}{\underset{\text{(TP)}}{\text{TRUE POSITIVES}} + \underset{\text{(FP)}}{\text{FALSE POSITIVES}}}$$

Contoh untuk confusion matrix berikut:

```
Confusion matrix:
[[397 103]
 [ 79 421]]
```

Maka perhitungan presisinya adalah:

Presisi = 397 / (397+103) = 0.794

Jadi, nilai presisinya adalah 79.4%.