

***SENTIMENT ANALYSIS PADA PERUSAHAAN YELP
MENGUNAKAN ALGORITMA MULTINOMIAL NAÏVE BAYES***



TUGAS UJIAN AKHIR SEMESTER 6 DATA MINING

Disusun Oleh :

1. Izzatur Royhan (16090067)
2. Dhiya Reksa K B (16090067)

**Politeknik Harapan Bersama Tegal
Tegal
2019**

1. JUDUL

“Sentimen analisis pada perusahaan yelp menggunakan algoritma *multinomial naïve bayes*”.

2. Pendahuluan

2.1. Latar Belakang

Yelp adalah perusahaan multinasional Amerika Serikat yang berkantor pusat di San Francisco, California. Perusahaan ini mengembangkan, menginangi, dan memasarkan Yelp.com dan aplikasi telepon Yelp. Yelp menerbitkan penilaian badan usaha di lingkungan sekitar pengguna dengan bantuan urun daya. Yelp juga memiliki layanan pemesanan daring Yelp Reservations dan layanan pengiriman makanan daring Eat24. Perusahaan ini juga melatih UKM untuk membalas penilaian pengunjung, mengadakan acara sosial untuk para pengunjung, dan menyediakan data tentang badan usaha tersebut seperti skor inspeksi kesehatan.

Yelp didirikan tahun 2004 oleh mantan karyawan PayPal Russel Simmons dan Jeremy Stoppelman. Yelp tumbuh besar dalam waktu singkat dan mendapat kucuran dana yang banyak. Pada tahun 2010, Yelp meraup pendapatan sebesar \$30 juta dan situsnya mengumpulkan 4,5 juta penilaian pengunjung. Sejak tahun 2009 sampai 2012, Yelp memperluas wilayah operasinya di Eropa dan Asia. Tahun 2009, Yelp merundingkan akuisisi dengan Google. Yelp menjadi perusahaan terbuka pada bulan Maret 2012 dan meraup untung untuk pertama kalinya dua tahun kemudian. Per 2016, Yelp.com dikunjungi oleh 135 juta orang setiap bulan dan memiliki 95 juta penilaian pengunjung. Pendapatan perusahaan berasal dari iklan bisnis.

Untuk mengetahui reaksi dari pengguna aplikasi yelp di twitter, sehingga kita memerlukan data dari twitter dengan menggunakan teknik crawling, python sebagai bahasa pemrograman untuk crawling.

2.2. Tujuan Penelitian

Adapun tujuan dilakukan penelitian adalah Mengolah dataset yelp_labelled menjadi model prediksi menggunakan algoritma multinomial naïve bayes untuk bisa membedakan tweet itu positif atau negatif serta menganalisa setiap reaksi pengguna twitter terhadap perusahaan yelp.

2.3. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat bagi :

- a. Menambah ilmu, pengalaman, dan pengetahuan pada bidang Teknologi Informasi khususnya dalam konsep data mining.
- b. Mengetahui sebuah tweet bersifat positif atau negatif dengan persentase akurasi yang besar karena menggunakan model prediction dari dataset berlabel aktual.

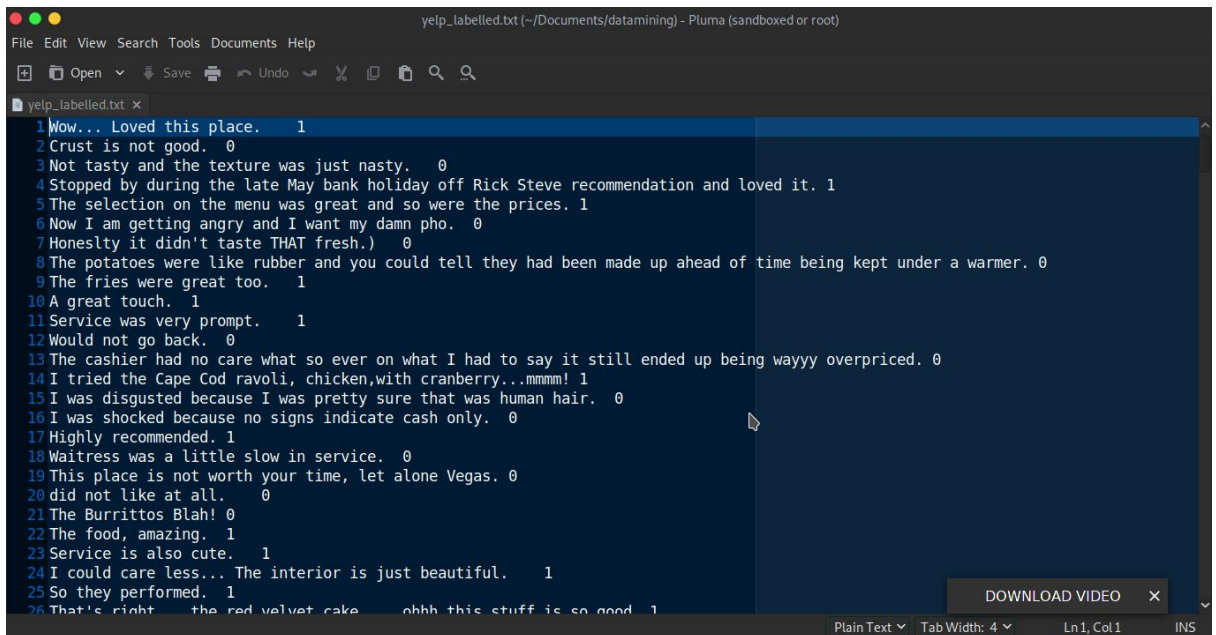
3. Pembahasan

1. Persiapkan dataset dan library yang dibutuhkan

Library yang kita butuhkan:

- pandas
- Tweepy
- Nltk
- Sklearn

Buat terlebih dahulu file dengan nama app.py, twitter.py, processing.py, twitter.py dan gui.py kemudian download dataset yelp_labelled.txt.



Dataset yelp_labelled yang telah di download

2. Membuat class processing untuk cleaning data

Masuk kedalam file processing.py kemudian edit seperti dibawah ini :

```
import re
import string
from string import punctuation
from nltk.corpus import stopwords

class Preprocessing :
    def __init__(self):
        print("Initializing preprocessing...")
        pass

    def processTweet(self, tweet):
        tweet = re.sub(r'\&\w*;', ' ', tweet)
```

```

tweet = re.sub('@[\s]+', "", tweet)
tweet = re.sub(r'\$w*', "", tweet)
tweet = tweet.lower()
tweet = re.sub(r'https?:\V.*\w*', "", tweet)
tweet = re.sub(r'#\w*', "", tweet)
tweet = re.sub(r'[' + punctuation.replace('@', '') + ']+', '', tweet)
tweet = re.sub(r'\b\w{1,2}\b', "", tweet)
tweet = re.sub(r'\s\s+', ' ', tweet)
tweet = tweet.lstrip(' ')
tweet = "".join(c for c in tweet if c <= '\uFFFF')
return tweet

```

```

def text_process(self, raw_text):
    nopunc = [char for char in list(raw_text) if char not in
string.punctuation]
    nopunc = "".join(nopunc)
    return [word for word in nopunc.lower().split() if word.lower() not in
stopwords.words('english')]

```

Fungsi dari source code diatas adalah untuk membersihkan data yang diperoleh dari simbol-simbol dan karakter yang tidak diperlukan.

Selanjutnya kita buka dan edit twitter.py lalu masukan kode seperti dibawah ini :

```

import tweepy

class Twitter :
    def __init__(self):
        print("hehe")
        pass

    def instance(self):

```

#Gantilah consumer key, secret dan access key/secret sesuai dengan milik anda, jika belum punya, anda bisa mengajukan ke twitter

```
CONSUMER_KEY = "ARhmOYK2f5xR5PGCtUPZwNayu"
CONSUMER_SECRET =
"Pt6Z55l8RynOvEcLyYtxR1uarOKhar939YnsVI0S7OcLZjGPtE"
ACCESS_KEY =
"1031914262896074759-UKdpwFlj8Ylw3zLXjINAjIEwb2VoSi"
ACCESS_SECRET =
"JYJenkiJP6UEzoQljCikqWb0clmGoyTG6uz8xZUpDpXD1"
api = tweepy.OAuthHandler(consumer_key = CONSUMER_KEY,
consumer_secret = CONSUMER_SECRET)
api.set_access_token(ACCESS_KEY, ACCESS_SECRET)
return tweepy.API(api, wait_on_rate_limit=True,
wait_on_rate_limit_notify=True)
```

Fungsi dari source code diatas adalah untuk mendapatkan akses API dari twitter.

Selanjutnya kita buka dan edit crawler.py lalu masukan kode seperti dibawah ini :

```
import tweepy
import time
from twitter import Twitter
from preprocessing import Preprocessing
import csv
API = Twitter().instance()
waitQuery = 100
waitTime = 2.0
engineBlow = 1
Preprocessing = Preprocessing()
csvFile = open('yelp_from_twitter.csv', 'w', encoding='utf-8')
csvWriter = csv.writer(csvFile)
```

```

def search() :
    global API, waitQuery, waitTime, engineBlow
    query = str(input("Search something : "))
    total_number = int(input("n : "))
    cursor = tweepy.Cursor(API.search, query + " -RT", tweet_mode =
"extended", lang = "en").items()
    count = 0
    error = 0
    secondcount = 0
    while secondcount < total_number:
        try:
            c = next(cursor)
            count += 1
            if count % waitQuery == 0:
                time.sleep(waitTime)
        except tweepy.TweepError:
            print("Sleeping...")
            time.sleep(60 * engineBlow)
            c = next(cursor)
        except StopIteration:
            break

    try:
        text_val = c._json['full_text']
        text_val = str(text_val).lower()
        text_val = Preprocessing.processTweet(text_val)
        if "rt" not in text_val:
            if len(text_val) != 0:
                secondcount += 1
                csvWriter.writerow([secondcount, str(text_val)])
                print("[INFO] Getting a tweet : " + str(secondcount) + " = " +
text_val)
    except Exception as e:

```

```
error += 1
print('[EXCEPTION] Stream data: ' + str(e))
```

```
search()
```

Fungsi dari source code diatas adalah untuk mengambil data dari twitter.

Selanjutnya kita buka dan edit app.py lalu masukan kode seperti dibawah ini :

```
import pandas as pd
from sklearn.pipeline import Pipeline
from preprocessing import Preprocessing
from sklearn.externals import joblib
import nltk
import csv
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
from sklearn.naive_bayes import MultinomialNB

yelp_dataset = pd.read_csv("yelp_labelled.txt", sep="\t", header=None)
yelp_dataset.columns = ['text', 'label']
positives = yelp_dataset['label'][yelp_dataset.label == 1]
negatives = yelp_dataset['label'][yelp_dataset.label == 0]
COLNAMES = ["id", "text"]
nltk.download('stopwords')
```



```
def word_count(text):  
    return len(str(text).split())
```

```
yelp_dataset["word_count"] = yelp_dataset["text"].apply(word_count)  
print("Dataset loaded successfully!")
```

```
all_words = []  
for line in list(yelp_dataset["text"]):  
    words = line.split()  
    for word in words:  
        all_words.append(word.lower())
```

```
dataset = yelp_dataset
```

```
####
```

```
dataset.to_pickle("dataset.p")  
dataset_pickle = pd.read_pickle("dataset.p")  
dataset_pickle["text"] =  
dataset_pickle["text"].apply(Preprocessing().processTweet)  
dataset_pickle_pickle = dataset_pickle.drop_duplicates('text')  
dataset_pickle.shape
```

```
eng_stop_words = stopwords.words('english')  
dataset_pickle = dataset_pickle.copy()  
dataset_pickle["tokens"] =  
dataset_pickle["text"].apply(Preprocessing().text_process)
```

```
bow_transformer =  
CountVectorizer(analyzer=Preprocessing().text_process).fit(dataset_pickle["text"])
```

```

messages_bow = bow_transformer.transform(dataset_pickle['text'])
# print('Shape of Sparse Matrix: ', messages_bow.shape)
# print('Amount of Non-Zero occurrences: ', messages_bow.nnz)
print("Dataset dibersihkan!")
print("\nMulai train / test dengan perbandingan training 80% dan testing
20%")
# test nya hanya 20%, training nya 80%
X_train, X_test, y_train, y_test = train_test_split(yelp_dataset['text'],
yelp_dataset['label'], test_size=0.2)

pipeline = Pipeline([('bow', CountVectorizer(strip_accents='ascii',
stop_words='english', lowercase=True)),('tfidf', TfidfTransformer()),
('classifier', MultinomialNB()), ])

parameters = {'bow__ngram_range': [(1, 1), (1, 2)], 'tfidf__use_idf':
(True, False), 'classifier__alpha': (1e-2, 1e-3), }

grid = GridSearchCV(pipeline, cv=10, param_grid=parameters,
verbose=1)
grid.fit(X_train, y_train)

# hasil ->
# print("\nModel: %f using %s" % (grid.best_score_,
grid.best_params_))
# print("\n")
means = grid.cv_results_['mean_test_score']
stds = grid.cv_results_['std_test_score']
params = grid.cv_results_['params']
# for mean, stdev, param in zip(means, stds, params):
#     print("Mean: %f Stdev:(%f) with: %r" % (mean, stdev, param))

joblib.dump(grid, "model.pkl")
# buat test model

```

```
model_NB = joblib.load("model.pkl")
```

```
y_preds = model_NB.predict(X_test)
```

```
print('akurasi dari train/test split: ', str(accuracy_score(y_test, y_preds) *  
100) + "%")
```

```
print('confusion matrix: \n', confusion_matrix(y_test, y_preds))
```

```
print(classification_report(y_test, y_preds))
```

```
# testing
```

```
model_NB = joblib.load("model.pkl")
```

```
def label_to_str(x):
```

```
    if x == 0:
```

```
        return 'Negative'
```

```
    else:
```

```
        return 'Positive'
```

```
x = 0
```

```
text_ = [0] * len(yelp_dataset)
```

```
label_ = [0] * len(yelp_dataset)
```

```
for review in yelp_dataset['text']:
```

```
    predict = model_NB.predict([review])
```

```
    text_[x] = review
```

```
    label_[x] = predict[0]
```

```
    x += 1
```

```
print("write ke csv")
```

```
hehe = {"text": text_, "label": label_}
```

```
hehe2 = pd.DataFrame(data=hehe)
```

```

hehe2.to_csv('test_ulang_dataset.csv', header=True, index=False,
encoding='utf-8')
hasil_test_ulang = pd.read_csv("test_ulang_dataset.csv",
header='infer')
hasil_test_ulang.columns = ['text', 'label']
recheck_pos = hasil_test_ulang['label'][hasil_test_ulang.label == 1]
recheck_neg = hasil_test_ulang['label'][hasil_test_ulang.label == 0]
print("Hasil test ulang punya positive prediksi sebanyak : "+
str(len(recheck_pos)) +" dan negatif sebanyak " +
str(len(recheck_neg)))

```

```

i = 0
# iya -> iya
true_positive = 0
# iya -> ora
false_negative = 0
# ora -> ora
true_negative = 0
# ora -> iya
false_positive = 0
for predicted_label in hasil_test_ulang['label']:
    if yelp_dataset['label'][i] == 1:
        if predicted_label == 1:
            true_positive += 1
        else:
            false_negative += 1

    if yelp_dataset['label'][i] == 0:
        if predicted_label == 0:
            true_negative += 1
        else:
            false_positive += 1
    i += 1

```

```

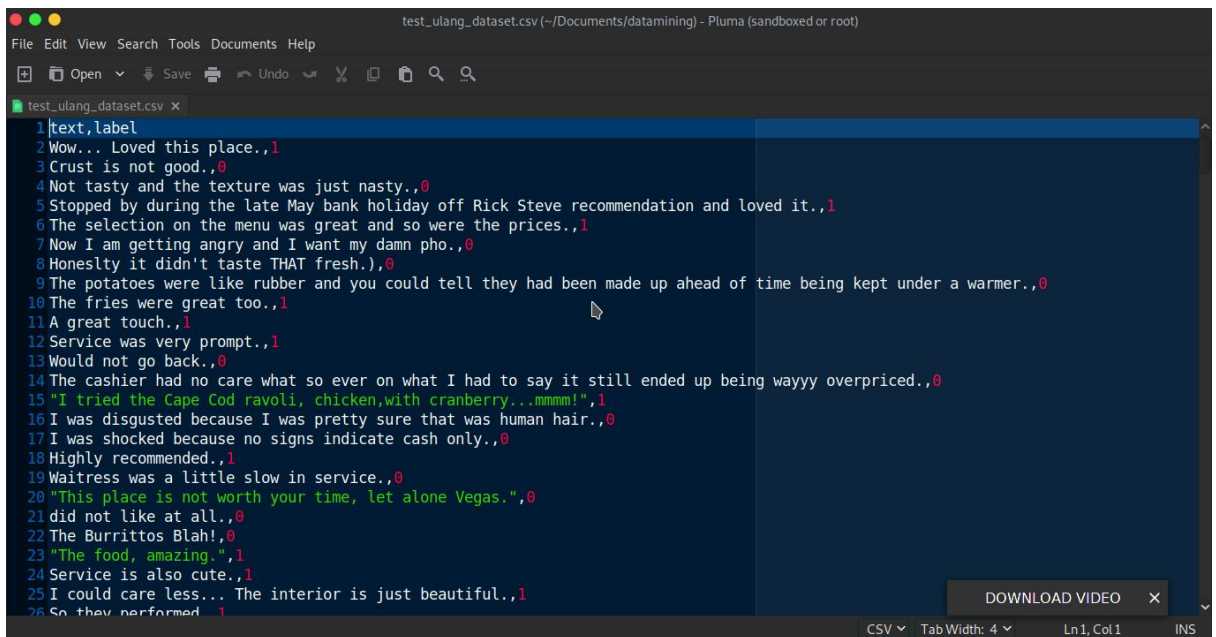
# print(yelp_dataset['label'] == predicted_label)
# print(predicted_label == 1)
print("True positive : " + str(true_positive))
print("True negative : " + str(true_negative))
print("False positive : " + str(false_positive))
print("False negative : " + str(false_negative))

# akurasi TP + TN / TP + FN + FP + TN
# menampilkan seluruh row (1000 row), berbeda dengan library yang
hanya mengambil sample 20% dr total row
print("Akurasi = " + str(
    ((true_positive + true_negative) / (true_positive + false_negative +
    false_positive + true_negative)) * 100) + "%")
print("Presisi = " + str((true_positive / (true_positive + false_positive)) *
    100) + "%")
print("Recall = " + str((true_positive / (true_positive + false_negative)) *
    100) + "%")
presisi = (true_positive / (true_positive + false_positive))
recall = (true_positive / (true_positive + false_negative))
print("f1-score = " + str(((presisi*recall)/(presisi+recall)*2) * 100) + "%")
print("#####")
print("#####")
print("test data from twitter")
from_twitter = pd.read_csv("yelp_from_twitter.csv")
from_twitter.columns = ['id', 'tweet']
csvFile = open('yelp_from_twitter_predict.csv', 'w', encoding='utf-8')
csvWriter = csv.writer(csvFile)
for tweet in list(from_twitter['tweet']):
    h = model_NB.predict([tweet])
    csvWriter.writerow([str(tweet), label_to_str(h[0])])
print("DONE!")

```

3. Menjalankan Program

- Pertama jalankan program crawler.py untuk mengambil data dari twitter terhadap perusahaan yelp
- Setelah data didapatkan kemudian jalankan app.py dengan perbandingan data 80% training data dan 20% data testing. Nilai classification_report yang anda terima adalah nilai dari 20% testing tadi. Setelah anda menjalankan app.py akan ada file baru yang dibuat bernama test_ulang_dataset.csv yang mengetest ulang seluruh row dari dataset yg kita punya (bukan 20% lagi), hasilnya :



```
1|text,label
2|Wow... Loved this place.,1
3|Crust is not good.,0
4|Not tasty and the texture was just nasty.,0
5|Stopped by during the late May bank holiday off Rick Steve recommendation and loved it.,1
6|The selection on the menu was great and so were the prices.,1
7|Now I am getting angry and I want my damn pho.,0
8|Honeslty it didn't taste THAT fresh.,0
9|The potatoes were like rubber and you could tell they had been made up ahead of time being kept under a warmer.,0
10|The fries were great too.,1
11|A great touch.,1
12|Service was very prompt.,1
13|Would not go back.,0
14|The cashier had no care what so ever on what I had to say it still ended up being wayyy overpriced.,0
15|"I tried the Cape Cod ravioli, chicken,with cranberry...mmmm!",1
16|I was disgusted because I was pretty sure that was human hair.,0
17|I was shocked because no signs indicate cash only.,0
18|Highly recommended.,1
19|Waitress was a little slow in service.,0
20|"This place is not worth your time, let alone Vegas.",0
21|did not like at all.,0
22|The Burritos Blah!,0
23|"The food, amazing.",1
24|Service is also cute.,1
25|I could care less... The interior is just beautiful.,1
26|So they performed ,1
```

4. Membuat model prediksi yang sudah dibuat

- Buat sebuah file dengan nama gui.py
Buka dan edit file gui.py seperti dibawah ini :

```
import tkinter as tk
import joblib
from tkinter import messagebox

model = joblib.load("model.pkl")
```

```

def label_to_str(x):
    if x == 0:
        return 'Negatif'
    else:
        return 'Positif'

def btn_event():
    text = editText.get()
    print()
    h = model.predict([text])
    hasil = label_to_str(h[0])
    tk.messagebox.showinfo("Hasil", "Kalimat yang anda masukkan
besentimen : "+hasil)
def hehe(text):
    print(text)

form = tk.Tk()
form.title("Royhan")

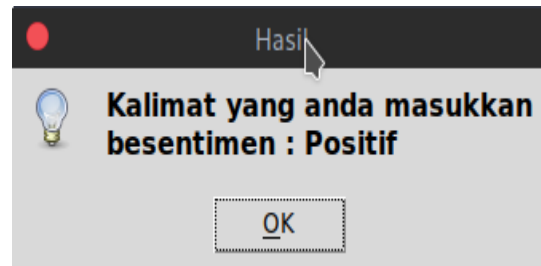
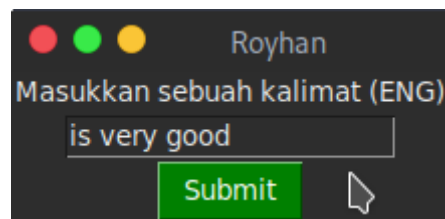
label1 = tk.Label(form, text = "Masukkan sebuah kalimat (ENG)")
editText = tk.Entry(form)
btn = tk.Button(form, text = "Submit", bg = "green", fg = "white",
command = lambda : btn_event())

label1.grid(row = 0)
editText.grid(row = 1)
btn.grid(row = 2)

form.mainloop()

```

- Jalankan gui.py dan hasilnya seperti dibawah ini :



Hasil dari kata yang diinputkan adalah positif

Berdasarkan dari 1000 row dataset yang kami punya (500 berlabel positif dan 500 berlabel negatif) dan dicocokkan dengan hasil prediksi menggunakan model yang sudah kami buat, kami menghitungnya dengan cara membuat script untuk mendapatkan nilai confusion matrix. Hasilnya seperti berikut:

```

Parrot Terminal
File Edit View Search Terminal Tabs Help
Parrot Terminal x Parrot Terminal x Parrot Terminal x

DeprecationWarning)
akurasi dari train/test split: 75.0%
confusion matrix:
[[68 15]
 [35 82]]

      precision    recall  f1-score   support

     0       0.66      0.82      0.73       83
     1       0.85      0.70      0.77      117

 accuracy          0.75
 macro avg          0.75
weighted avg          0.77

write ke csv
Hasil test ulang punya positive prediksi sebanyak : 482 dan negatif sebanyak 518
True positive : 462
True negative : 480
False positive : 20
False negative : 38
Akurasi = 94.19999999999999%
Presisi = 95.850622406639%
Recall = 92.4%
f1-score = 94.09368635437883%

```


4. KESIMPULAN Banyak algoritma yang dapat digunakan untuk melakukan sentiment analysis dan multinomial naïve bayes adalah salah satunya. Nilai keakuratan dari model prediksi yang dibuat sangat bergantung bagaimana data yang kita punya dan bagaimana cara kita memprosesnya. Teknik evaluasi dan visualisasi juga penting untuk mengukur bagaimana performa dari model yang sudah kita buat dan juga bagaimana sebuah kumpulan data yang besar dapat dipahami oleh orang awam.