Data Science Project

Penerapan Text Processing dalam Klasifikasi Sentimen Masyarakat di Twitter Menggunakan Algoritma SVM

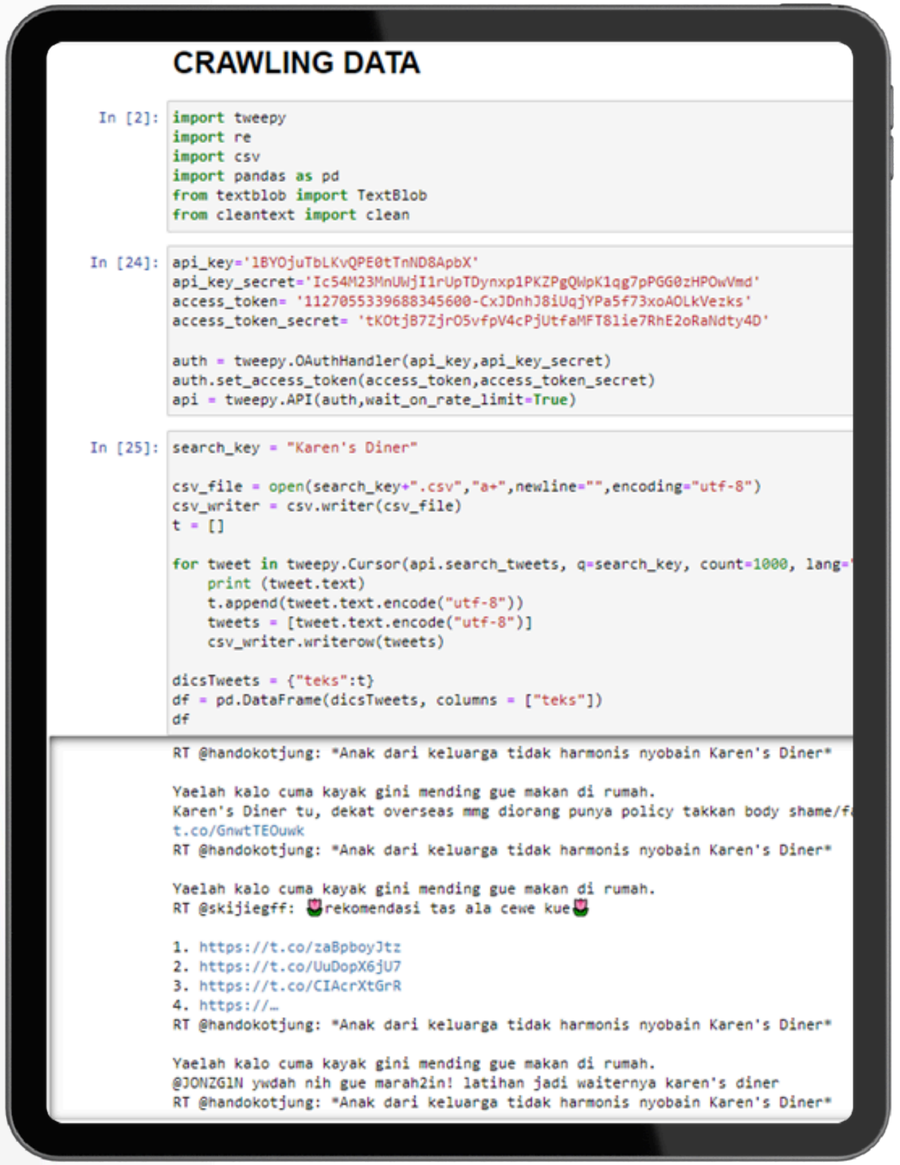
Studi Kasus : Karens Dinner

Table of Content

Crawling Data

Crawling data adalah proses pengambilan data yang tersedia secara online untuk umum. Pada kasus ini kami mengambil data dari twitter dengan memamnfaatkan Twitter API.

- Data yang diambil sebanyak 1000 data
- Data diambil pada 19 Desember 2022



Cleaning Data

Data Cleaning adalah suatu prosedur untuk memastikan kebenaran, konsistensi, dan kegunaan suatu data yang ada dalam dataset. Data Cleaning diperlukan agar teks lebih dapat dipahami.

Regular Expression python (import re) digunakan untuk memanipulasi (membersihkan) data yang di crawling

```
[10]: def CleanTxt(text):
        text = re.sub(r"@(\w+)", ' ', text)        #mentions
        text = re.sub('@[^\s]+','',text)           #tags
        text = re.sub('https?:\/\/\S+', '', text) #url
        text = re.sub('RT[\s]+', '', text)         #retweet
        text = re.sub(r'#', '', text)              #tanda #
        text = re.sub(r',', '', text)              #tanda ,
        text = re.sub(r'$', '', text)              #tanda $
        text = re.sub(r'-', '', text)              #tanda -
        text = re.sub(r'[^\w]|_', ' ', text)       #punctuations
        text = text.lower()                        #lowercase
        text = re.sub('[0-9]+', '', text)
        text = re.sub(r'b', '', text)
        text = re.sub(r'xf', '', text)
        text = re.sub(r'xc', '', text)
        text = re.sub(r'x', '', text)
        text = re.sub(r'xs', '', text)
        text = re.sub(r'xe', '', text)
        return text

      # Cleaning the text
      df['Tweets'] = df['Tweets'].apply(CleanTxt)
      df.drop_duplicates(subset ="Tweets", keep = 'first', inplace = True)
      df
```

t[10]:

	Tweets
0	anak dari keluarga tidak harmonis nyoain k...
1	karen s diner tu dekat overseas mmg diorang p...
3	rekomendasi tas ala cewe kue n n
5	ywdah nih gue marahin latihan jadi waitern...
7	tukang fotocopy depan komplek gue cocok nih...
...	...
973	karen s diner indo udah melenceng kejauhan ...
974	menko pmk muhadjir effendy mengingatkan mes...
975	ngapain ke karen s diner kalo mamak kau say...
979	ga perlu ke karen e s diner kemaren satu non...
995	ini yang kerja di karen s diner apa ndak ta...

232 rows × 1 columns

Labeling Data

Labeling Data adalah suatu prosedur melabel data untuk merepresentasikan karakter dari data tersebut. Label ini akan dijadikan atribut target untuk klasifikasi sentimen.

	A	B
1	Tweets	Label
2	tukang fotocopy depan komplek aku cocok pelayan karens diner	1
3	karens diner punya gofood nggak kalau punya gojek dimarahin waiter	0
4	orang medan masuk karens diner pelayan minggir	1
5	video karens diner paling asik	1
6	manager supervisor direktur karens diner tamu datang langsung dapat hissing	0
7	orang indonesia keseharian sopan santun karens diner jakarta akan gagal	1
8	cara karyawan karens diner indonesia mencaci pelanggan kesan cringe	0
9	seru jadi pelayan karens diner	1
10	emosi ngapain makan karens diner baca tag dcgallery cukup	0
11	nggak cocok orang indonesia sama konsep karens diner	0
12	ke karens diner conten creator bikin video upload	1
13	karens diner belum rasa didatangi ormas minta uang keamanan	1
14	karen bukan nggak cocok di indonesia netizen lebih gemas sebab konsep tidak begitu	0
15	fachrul ke karens diner waiters yang kena roasting	0
16	ide konten seleb tiktok datang ke karens diner mengamalkan falsafah jawa nrimo ing pandum	1
17	yang cocok jadi karyawan karens diner anak-anak dream	1
18	kaget teman aku jadi waiter karens diner	1
19	kakak dan ibu administrasi akademik cocok jadi pelayan karens diner	0
20	aku mau ngelamar part time karens diner teman bilang aku cocok kerja disana	1
21	intip pegawai karens diner mencakmencak	0
22	sudah cocok jadi pelayan karens diner belum	1
23	karens diner jakarta belajar sama tukang siomay kalimalang dan jimmy hendrix	1
24	review menu karens diner jakarta restoran viral pelayan judes dan kasar	1
25	kalau hantu sini maju tapi kalau bilang mukaku kayak pantat panci aku teraring	0
26	segini harga makan burger termahal di karens diner	1
27	nggak perlu repot ke karens diner cukup tinggal di medan	1
28	karens diner ada di jalan kalimantan jemer namanya judes	1
29	karyawan adiksi cocok kerja di karens diner	1

Preprocessing Data

Pre-processing adalah teknik untuk menyiapkan data agar lebih siap untuk dilakukan lebih lanjut dalam rangka ekstraksi pengetahuan.

```
#Tokenization

def tokenization(text):
    text = re.split('\W+',text)
    return text

df['Tokenization'] = df['Tweets'].apply(lambda x:tokenization(x.lower()))
df.head
```

```
#Stop Removal

stopword = nltk.corpus.stopwords.words('Indonesian')

def remove_stopwords(text):
    text = [word for word in text if word not in stopword]
    return text

df['Stop_Removal'] = df['Tokenization'].apply(lambda x: remove_stopwords(x))
df
```

```
#Normalizazion
normalizad_word = pd.read_csv('KarensDiner2.csv')

normalizad_word_dict = {}

for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[0]

def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]
df['Normalisasi'] = df['Stop_Removal'].apply(normalized_term)

df['Normalisasi'].head(10)
```

```
# ------ Proses Stemming --------
# import Sastrawi package
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter


# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# stemmed
def stemmed_wrapper(term):
    return stemmer.stem(term)

term_dict = {}

for document in df['Normalisasi']:
    for term in document:
        if term not in term_dict:
            term_dict[term] = ' '

print(len(term_dict))
print("------------------------")

for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)
    print(term,":" ,term_dict[term])

print(term_dict)
print("------------------------")


# apply stemmed term to dataframe
def get_stemmed_term(document):
    return [term_dict[term] for term in document]

df['Stemming'] = df['Normalisasi'].swifter.apply(get_stemmed_term)
print(df['Stemming'])
```

Pembuatan Model

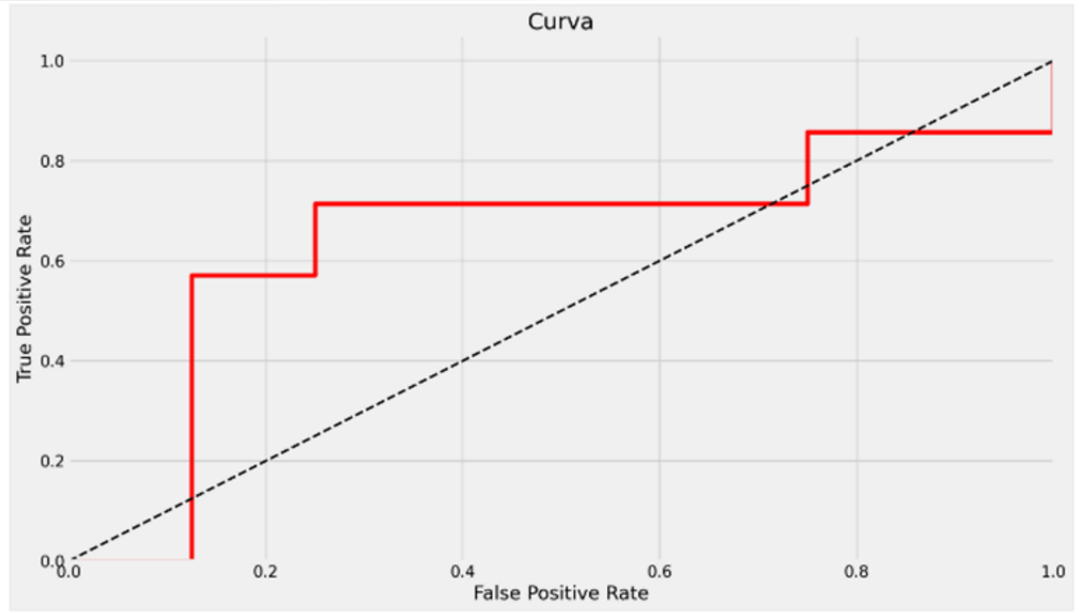
Algoritma yang dipakai untuk klasifikasi kali ini adalah SVM. SVM dipilih karena menawarkan akurasi tinggi dan bekerja dengan baik dengan ruang dimensi tinggi.

```
In [46]: x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.8, random_state=20)

In [77]: def tokenize(text):
             tknzr = TweetTokenizer()
             return tknzr.tokenize(text)

         def stem(doc):
             return (stemmer.stem(w) for w in analyzer(doc))

         vectorizer = CountVectorizer(
             analyzer = 'word',
             tokenizer = tokenize,
             lowercase = True,
             ngram_range=(1, 1))

         np.random.seed(1)

         pipeline_svm = make_pipeline(vectorizer, SVC(probability=True, kernel="linear", class_weight="balanced"

         grid_svm = GridSearchCV(pipeline_svm,param_grid = {'svc__C': [0.01, 0.1, 1]}, cv = kfolds, scoring="roc

         grid_svm.fit(X_train, y_train)
         grid_svm.score(X_test, y_test)

         Fitting 10 folds for each of 3 candidates, totalling 30 fits

Out[77]: 0.6428571428571428

In [78]: grid_svm.best_params_

Out[78]: {'svc__C': 1}

In [84]: grid_svm.best_score_

Out[84]: 0.8274659863945578
```

```
from sklearn.model_selection import learning_curve

train_sizes, train_scores, test_scores = \
    learning_curve(grid_svm.best_estimator_, X_train, y_train, cv=5, n_jobs=-1,
                   scoring="roc_auc", train_sizes=np.linspace(.1, 1.0, 10), random_state=1)
```

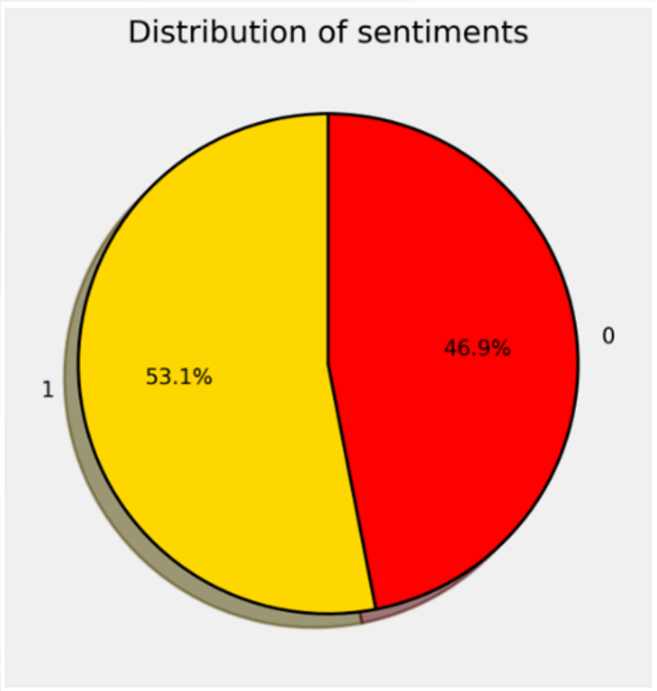
```
def plot_learning_curve(X, y, train_sizes, train_scores, test_scores, title='', ylim=None, figsize=(14,

    plt.figure(figsize=figsize)
    plt.title(title)
    if ylim is not None:
        plt.ylim(*ylim)
    plt.xlabel("Training examples")
    plt.ylabel("Score")

    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)
    plt.grid()

    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                     train_scores_mean + train_scores_std, alpha=0.1,
                     color="r")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                     test_scores_mean + test_scores_std, alpha=0.1, color="g")
    plt.plot(train_sizes, train_scores_mean, 'o-', color="r",
             label="Training score")
    plt.plot(train_sizes, test_scores_mean, 'o-', color="g",
             label="Cross-validation score")

    plt.legend(loc="lower right")
    return plt
```

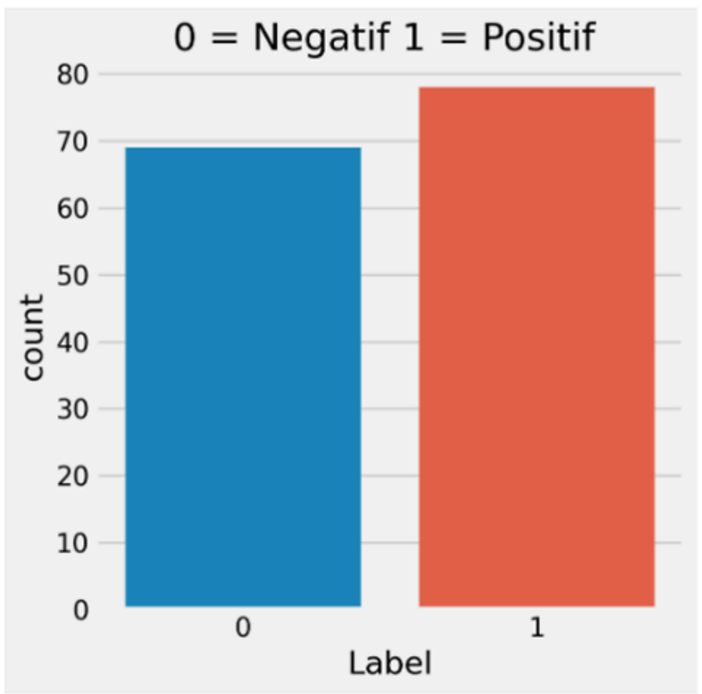
Pengujian Model




```
In [86]: grid_svm.predict(["Karens Diner pelayannya pemarah"])
Out[86]: array([0], dtype=int64)

In [81]: grid_svm.predict(["Karens Diner saat ini banyak pengunjung"])
Out[81]: array([1], dtype=int64)
```

Visualisasi Data





Thank You