

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

4.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 4.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

4.1.1 Teori

4.1.2 Praktek

4.1.3 Penanganan Error

4.1.4 Bukti Tidak Plagiat



Gambar 4.2 Kecerdasan Buatan.

4.2 1174069 - Fanny Shafira Damayanti

4.2.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.3 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri.

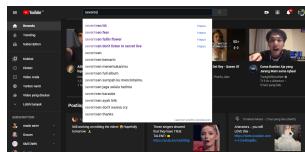
Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.4 Klasifikasi Bunga.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.

Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.5 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.

Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.6 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.7 TF-IDF.

4.2.2 Praktek Program

1. Soal 1

```

1 #%% Soal 1
2 import pandas as pd #digunakan untuk mengimport library
# pandas dengan alias pd
3 pd = pd.read_csv("F:/Semester 6/Artificial Intelligence/
# Tugas 4/src/csv_fanny.csv") #membaca file csv

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :

Gambar 4.8 Hasil Soal 1.

2. Soal 2

```

1 #%% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data
      yang tersedia

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :

Gambar 4.9 Hasil Soal 2.

3. Soal 3

```

1 #%% Soal 3
2 import pandas as fanny #untuk import library pandas berguna
      untuk mengelola dataframe
3 fanny = fanny.read_csv("F://Semester 6/Artificial
      Intelligence/Tugas 4/src/Youtube03-LMFAO.csv") #membaca
      file dengan format csv
4
5 spam=fanny.query('CLASS == 1') #membagi tabel spam
6 nospam=fanny.query('CLASS == 0')#membagi tabel no spam
7
8 from sklearn.feature_extraction.text import CountVectorizer #
      untuk import countvectorizer berfungsi untuk memecah data
      tersebut menjadi sebuah kata yang lebih sederhana
9 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
      tersebut , pada code ini tidak ada hasilnya dikarenakan
      spyder tidak mendukung hasil dari instasiasi.
10
11 dvec = vectorizer.fit_transform(fanny['CONTENT']) #untuk
      melakukan pemecahan data pada dataframe yang terdapat pada
      kolom konten

```

```

12 dvec #Untuk menampilkan hasil dari code sebelumnya
13
14 Daptarkata= vectorizer.get_feature_names()
15
16 dshuf = fanny.sample(frac=1)
17
18 d_train=dshuf[:300]
19 d_test=dshuf[300:]
20
21 d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
22 d_train_att
23
24 d_train_label=d_train['CLASS']
25 d_test_label=d_test['CLASS']

```

Dengan menggunakan $1174069 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

| fanny - DataFrame | | | | | |
|-------------------|-----------------|---------------|------------------|----------------------------------|-------|
| Index | COMMENT_ID | AUTHOR | DATE | CONTENT | CLASS |
| 0 | r13au1nephew | Cory Wilson | 2015-09-28T23... | ca http:// | 0 |
| 1 | r124yvrcxat | Spike Gaming | 2015-09-28T23... | view out / | 0 |
| 2 | r13ttc3jy5JN... | Lsd Music | 2015-09-28T23... | Hey guys I'm back I'm here | 1 |
| 3 | r13t3m0npma... | Oney3 Fox | 2015-09-28T23... | Party | 0 |
| 4 | r13evcixekw... | PATRICK_TV | 2015-09-28T23... | Party rock | 0 |
| 5 | r13nxlymew... | Brian Brasl | 2015-09-28T0... | Shutup! | 0 |
| 6 | r13betkll0ov... | Brian Brasl | 2015-09-28T0... | Omg | 0 |
| 7 | r133i0n0odqu... | Alex Defeo | 2015-09-28T0... | This song is just sick | 0 |
| 8 | r13g6n6ek... | Glowand... | 2015-09-28T0... | you never >_< | 0 |
| 9 | r13lmmcr0n... | Silvia Basso | 2015-09-28T0... | wow!!!!!! | 0 |
| 10 | r13ctyFv4n... | Michael | 2015-09-28T0... | I love this | 0 |
| 11 | r13qzqfj5n... | whitehorse | 2015-09-28T0... | song so much | 0 |
| 12 | r13qzqfj5n... | rodrigued | 2015-09-28T0... | I kiss when | 0 |
| 13 | r13tr3vztr... | Alex Martin | 2015-09-28T0... | people dress, | 0 |
| 14 | r13lZm2ztrv... | Alex Jansen | 2015-09-28T0... | last year or, | 0 |
| 15 | r13mkfz3koy... | Joe Miller | 2015-09-27T23... | ever!!!! | 0 |
| 16 | r13t97jx1n... | Joe Miller | 2015-09-27T23... | love music | 0 |
| 17 | r13qzqfj5n... | vlad kerec | 2015-09-27T23... | emo/emo rock | 0 |
| 18 | r13t97jx1n... | SoulInDust... | 2015-09-27T23... | (S) | 0 |
| 19 | r13t97jx1n... | Nguchi Nguci | 2015-09-27T23... | you watched .. | 0 |
| 20 | r13wq3q000... | Ferrari | 2015-09-27T23... | incredible | 0 |

Gambar 4.10 Hasil Soal 3.

4. Soal 4

```

1 %% Soal 4
2
3 from sklearn import svm
4 clfsvm = svm.SVR(gamma = 'auto')
5 clfsvm.fit(d_train_att , d_train_label)

```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```

In [7]: from sklearn import svm
...: clfsvm = svm.SVR(gamma = 'auto')
...: clfsvm.fit(d_train_att , d_train_label)
Out[7]:
SVR(C=1.0, cache_size=200, cache_size=200, cache_size=200, cache_size=200,
     class_weight=None, degree=3, epsilon=0.1,
     gamma='rbf', kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
     verbose=False)

```

Gambar 4.11 Hasil Soal 4.

5. Soal 5

```

1 #%%soal 5
2
3 from sklearn import tree
4 clftree = tree.DecisionTreeClassifier()
5 clftree.fit(d_train_att, d_train_label)

```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib.
Hasilnya adalah sebagai berikut :

```

In [8]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
Out[8]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None,
min_impurity_decrease=0.0,
min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated',
random_state=None, splitter='best')

```

Gambar 4.12 Hasil Soal 5.

6. Soal 6

```

1 #%%soal 6/
2
3 from sklearn.metrics import confusion_matrix
4 pred_labels=clftree.predict(d_test)
5 cm=confusion_matrix(d_test_label, pred_labels)
6
7 #%%
8
9 import matplotlib.pyplot as plt
10
11 def plot_confusion_matrix(cm, classes,
12                           normalize=False,
13                           title='Confusion matrix',
14                           cmap=plt.cm.Blues):
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
18         newaxis]
19         print("Normalized confusion matrix")
20     else:
21         print('Confusion matrix, without normalization')
22
23     print(cm)

```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:     """
...:     If normalize is True, cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:     print(cm)
...:     cm
```

Gambar 4.13 Hasil Soal 6.

7. Soal 7

```
1 %%soal 7
2
3 from sklearn.model_selection import cross_val_score
4
5 scores=cross_val_score(clf, d_train_att, d_train_label, cv=5)
6
7 skor_rata2=scores.mean()
8 skoresd=scores.std()
```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```
In [12]: from sklearn.model_selection import cross_val_score
...
...: scores=cross_val_score(clf, d_train_att, d_train_label, cv=5)
...: skor_rata2=scores.mean()
...: skoresd=scores.std()
```

Gambar 4.14 Hasil Soal 7.

8. Soal 8

```
1 %%soal 8 /
2
3 max_features_opts = range(5, 50, 5) #max_features_opts
4 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
5 rf_params = fanny.empp((len(max_features_opts)*len(
6     n_estimators_opts),4), float) #rf_params sebagai variabel
7     untuk menjumlahkan yang sudah di tentukan sebelumnya
8 i = 0
9 for max_features in max_features_opts: #pengulangan
10    for n_estimators in n_estimators_opts: #pengulangan
11        clf = RandomForestClassifier(max_features=
12            max_features, n_estimators=n_estimators) #menampilkan
13            variabel csf
14            scores = cross_val_score(clf, df_train_att,
15            df_train_label, cv=5) #scores sebagai variabel training
```

```

11         rf_params[i,0] = max_features #index 0
12         rf_params[i,1] = n_estimators #index 1
13         rf_params[i,2] = scores.mean() #index 2
14         rf_params[i,3] = scores.std() * 2 #index 3
15         i += 1 #dengan ketentuan i += 1
16         print("Max features: %d, num estimators: %d, accuracy
   : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
mean(), scores.std() * 2))
17     #print hasil pengulangan yang sudah ditentukan

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```

In [2]: Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 25, num estimators: 105, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 25, num estimators: 110, accuracy: 0.40 (+/- 0.02)
Out[2]: Max features: 25, num estimators: 115, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 25, num estimators: 120, accuracy: 0.41 (+/- 0.03)
Out[2]: Max features: 25, num estimators: 125, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 25, num estimators: 130, accuracy: 0.44 (+/- 0.03)
Out[2]: Max features: 25, num estimators: 135, accuracy: 0.44 (+/- 0.03)
In [2]: Max features: 25, num estimators: 140, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 25, num estimators: 145, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 25, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 25, num estimators: 155, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 25, num estimators: 160, accuracy: 0.46 (+/- 0.04)
Out[2]: Max features: 25, num estimators: 165, accuracy: 0.45 (+/- 0.04)
In [2]: Max features: 25, num estimators: 170, accuracy: 0.46 (+/- 0.04)
Out[2]: Max features: 25, num estimators: 175, accuracy: 0.46 (+/- 0.04)
In [2]: Max features: 25, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Out[2]: Max features: 25, num estimators: 185, accuracy: 0.46 (+/- 0.04)
In [2]: Max features: 25, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Out[2]: Max features: 25, num estimators: 195, accuracy: 0.46 (+/- 0.04)
In [2]: Max features: 25, num estimators: 200, accuracy: 0.46 (+/- 0.04)
Out[2]: Max features: 30, num estimators: 100, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 30, num estimators: 105, accuracy: 0.40 (+/- 0.02)
Out[2]: Max features: 30, num estimators: 110, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 30, num estimators: 115, accuracy: 0.41 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 120, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 30, num estimators: 125, accuracy: 0.44 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 130, accuracy: 0.44 (+/- 0.03)
In [2]: Max features: 30, num estimators: 135, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 140, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 30, num estimators: 145, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 150, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 30, num estimators: 155, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 160, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 30, num estimators: 165, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 170, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 30, num estimators: 175, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 180, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 30, num estimators: 185, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 190, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 30, num estimators: 195, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 30, num estimators: 200, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 105, accuracy: 0.40 (+/- 0.02)
In [2]: Max features: 35, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 115, accuracy: 0.41 (+/- 0.03)
In [2]: Max features: 35, num estimators: 120, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 125, accuracy: 0.44 (+/- 0.03)
In [2]: Max features: 35, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 135, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 35, num estimators: 140, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 145, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 35, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 155, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 160, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 165, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 175, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 180, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 185, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 35, num estimators: 195, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 35, num estimators: 200, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 100, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 40, num estimators: 105, accuracy: 0.40 (+/- 0.02)
Out[2]: Max features: 40, num estimators: 110, accuracy: 0.43 (+/- 0.03)
In [2]: Max features: 40, num estimators: 115, accuracy: 0.41 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 120, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 40, num estimators: 125, accuracy: 0.44 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 130, accuracy: 0.44 (+/- 0.03)
In [2]: Max features: 40, num estimators: 135, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 140, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 40, num estimators: 145, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 40, num estimators: 155, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 160, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 40, num estimators: 165, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 170, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 40, num estimators: 175, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 180, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 40, num estimators: 185, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 190, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 40, num estimators: 195, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 40, num estimators: 200, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 105, accuracy: 0.40 (+/- 0.02)
In [2]: Max features: 45, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 115, accuracy: 0.41 (+/- 0.03)
In [2]: Max features: 45, num estimators: 120, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 125, accuracy: 0.44 (+/- 0.03)
In [2]: Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 135, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 45, num estimators: 140, accuracy: 0.45 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 145, accuracy: 0.45 (+/- 0.03)
In [2]: Max features: 45, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 155, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 160, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 165, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 175, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 180, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 185, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Out[2]: Max features: 45, num estimators: 195, accuracy: 0.46 (+/- 0.03)
In [2]: Max features: 45, num estimators: 200, accuracy: 0.46 (+/- 0.03)

```

Gambar 4.15 Hasil Soal 8.

4.2.3 Penanganan Error

1. ScreenShot Error

```

FileNotFoundError: [Errno 2] File 'b'f1//Semester 6/Artificial Intelligence//tags 4//src/fanny.csv' does not exist: 'b'f1//Semester 6/Artificial Intelligence//tags 4//src/fanny.csv'

```

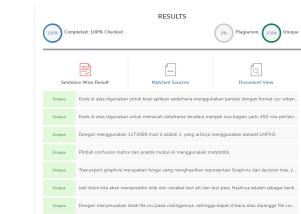
Gambar 4.16 SyntaxError

2. Cara Penanganan Error

- SyntaxError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.2.4 Bukti Tidak Plagiat



Gambar 4.17 Bukti Tidak Melakukan Plagiat Chapter 4

4.2.5 Link Youtube

<https://youtu.be/X-xd9Nb78Gs>

4.3 1174070 - Arrizal Furqona Gifary

4.3.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



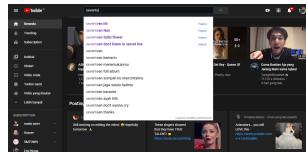
Gambar 4.18 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri. Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.19 Klasifikasi Bunga.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.
Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.20 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.
Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.
5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.21 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.
TF-IDF merupakan metode untuk menghitung bobot setiap kata pada

suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.22 TF-IDF.

4.3.2 Praktek Program

1. Soal 1

```

1 %% Soal 1
2 import pandas as pd #digunakan untuk mengimport library
# pandas dengan alias pd
3 pd = pd.read_csv("F:/Semester 6/Artificial Intelligence/
#Tugas 4/src/csv_izal.csv") #membaca file csv

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :



Gambar 4.23 Hasil Soal 1.

2. Soal 2

```

1 %% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data
# yang tersedia

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :



Gambar 4.24 Hasil Soal 2.

3. Soal 3

```

1 %% Soal 3
2 import pandas as izal #untuk import library pandas berguna
# untuk mengelola dataframe
3 izal = izal.read_csv("F:// Semester 6/Artificial Intelligence/
# Tugas 4/src/Youtube03-LMFAO.csv") #membaca file dengan
# format csv
4
5 spam=izal.query('CLASS == 1') #membagi tabel spam
6 nospam=izal.query('CLASS == 0')#membagi tabel no spam
7
8 from sklearn.feature_extraction.text import CountVectorizer #
# untuk import countvectorizer berfungsi untuk memecah data
# tersebut menjadi sebuah kata yang lebih sederhana
9 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
# tersebut , pada code ini tidak ada hasilnya dikarenakan
# spyder tidak mendukung hasil dari instasiasi.
10
11 dvec = vectorizer.fit_transform(izal[ 'CONTENT' ]) #untuk
# melakukan pemecahan data pada dataframe yang terdapat pada
# kolom konten
12 dvec #Untuk menampilkan hasil dari code sebelumnya
13
14 Daptarkata= vectorizer.get_feature_names()
15
16 dshuf = izal.sample(frac=1)
17
18 d_train=dshuf[:300]
19 d_test=dshuf[300:]
20
21 d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT' ])
22 d_train_att
23
24 d_train_label=d_train[ 'CLASS' ]
25 d_test_label=d_test[ 'CLASS' ]

```

Dengan menggunakan $1174070 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

| Index | COMMENT_ID | AUTHOR | DATE | CONTENT | CLASS |
|-------|----------------|----------------|--------------------|-----------------------------|-------|
| 0 | 1130w0shepd. | Corey Wilson | 2015-05-28T23...4a | ...ing me...ngt...erful but | 0 |
| 1 | 1124yrcxzaJb. | Felic Geling | 2015-05-28T23... | hey guys... Party rock | 1 |
| 2 | 1137zC5yfjBh. | Lek Music | 2015-05-28T23... | Rock...lol... | 0 |
| 3 | 1137zDm9pva. | Cheryl Fox | 2015-05-28T23... | Part...me is hu... | 0 |
| 4 | 1138px4vKsd. | PANTICO_7U | 2015-05-28T23... | Rock...lol... | 0 |
| 5 | 1139nyewm... | Brian Brad | 2015-05-28T0... | Surprise | 0 |
| 6 | 1139ed11oy... | Brian Brad | 2015-05-28T0... | 0 | 0 |
| 7 | 11391t0m0g... | Alex Reiko | 2015-05-28T0... | This song is | 0 |
| 8 | 1139mg8chL... | Gloward | 2015-05-28T0... | Just really =) | 0 |
| 9 | 1139emzcm... | Jessnet | 2015-05-28T0... | awesomeness /> | 0 |
| 10 | 1139emzcm... | Silvia Basilio | 2015-05-28T0... | Incredible song | 0 |
| 11 | 1139ctvFnd... | Unikide | 2015-05-28T0... | song so much | 0 |
| 12 | 1139g4k5u... | Notrigged | 2015-05-28T0... | 2015 Lifef... | 0 |
| 13 | 1139trxtetP... | Alex Martin | 2015-05-28T0... | people dress, | 0 |
| 14 | 1139uh2ztrv... | Alex Jensen | 2015-05-28T0... | last year of, | 0 |
| 15 | 1139w1x0my... | Adam Hill | 2015-05-27T2... | Best song | 0 |
| 16 | 1139trxtq1... | [Joe pebb... | 2015-05-27T2... | super nice, | 0 |
| 17 | 1139trxtq1... | Silvia Basilio | 2015-05-27T2... | super nice, | 0 |
| 18 | 1139trxtq1... | SoulInTheHorn | 2015-05-27T1... | PARTY ROCK | 0 |
| 19 | 1139trxtq1... | Phuong Linh | 2015-05-27T1... | Thumbs up if | 0 |
| 20 | 1139trxtq1... | Genitalo | 2015-05-27T1... | this is | 0 |
| | | | | | |

Gambar 4.25 Hasil Soal 3.

4. Soal 4

```
1 #%% Soal 4
2
3 from sklearn import svm
4 clfsvm = svm.SVR(gamma = 'auto')
5 clfsvm.fit(d_train_att , d_train_label)
```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```
In [7]: from sklearn import svm
... clfsvm = svm.SVR(gamma = 'auto')
... clfsvm.fit(d_train_att , d_train_label)
Out[7]: SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1,
gamma='auto', kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
verbose=False)
```

Gambar 4.26 Hasil Soal 4.

5. Soal 5

```
1 #%%soal 5
2
3 from sklearn import tree
4 clftree = tree.DecisionTreeClassifier()
5 clftree.fit(d_train_att , d_train_label)
```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut :

```
In [8]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
Out[8]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated', random_state=None, splitter='best')
```

Gambar 4.27 Hasil Soal 5.

6. Soal 6

```
1 #%%soal 6/
2
3 from sklearn.metrics import confusion_matrix
4 pred_labels=clftree.predict(d_test)
5 cm=confusion_matrix(d_test_label ,pred_labels)
6
7 #%%
8
9 import matplotlib.pyplot as plt
10
11 def plot_confusion_matrix(cm, classes,
12                           normalize=False,
13                           title='Confusion matrix',
14                           cmap=plt.cm.Blues):
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
18         print("Normalized confusion matrix")
19     else:
20         print('Confusion matrix, without normalization')
21
22     print(cm)
```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...:
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:     cm
```

Gambar 4.28 Hasil Soal 6.

7. Soal 7

```
1 #%%soal 7
2
```

```

3 from sklearn.model_selection import cross_val_score
4
5 scores=cross_val_score(clftree,d_train_att,d_train_label, cv
6 =5)
7
8 skor_rata2=scores.mean()
9 skoresd=scores.std()

```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```

In [12]: from sklearn.model_selection import cross_val_score
...:
scores=cross_val_score(clftree,d_train_att,d_train_label, cv=5)
...:
skor_rata2=scores.mean()
...:
skoresd=scores.std()

```

Gambar 4.29 Hasil Soal 7.

8. Soal 8

```

1 %%soal 8 /
2
3 max_features_opts = range(5, 50, 5) #max_features_opts
4 sebagai variabel untuk membuat range 5,50,5
5 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
6 sebagai variabel untuk membuat range 10,200,20
7 rf_params = izal.empty((len(max_features_opts)*len(
8     n_estimators_opts),4), float) #rf_params sebagai variabel
9 untuk menjumlahkan yang sudah di tentukan sebelumnya
10 i = 0
11 for max_features in max_features_opts: #pengulangan
12     for n_estimators in n_estimators_opts: #pengulangan
13         clftree = RandomForestClassifier(max_features=
14             max_features, n_estimators=n_estimators) #menampilkan
15         variabel csf
16         scores = cross_val_score(clf, df_train_att,
17             df_train_label, cv=5) #scores sebagai variabel training
18         rf_params[i,0] = max_features #index 0
19         rf_params[i,1] = n_estimators #index 1
20         rf_params[i,2] = scores.mean() #index 2
21         rf_params[i,3] = scores.std() * 2 #index 3
22         i += 1 #dengan ketentuan i += 1
23         print("Max features: %d, num estimators: %d, accuracy
24 : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
25 mean(), scores.std() * 2))
26         #print hasil pengulangan yang sudah ditentukan

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```

Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 100, accuracy: 0.33 (+/- 0.02)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 30, num estimators: 50, accuracy: 0.44 (+/- 0.03)
Max features: 30, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 30, num estimators: 100, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 150, accuracy: 0.46 (+/- 0.04)
Max features: 30, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.34 (+/- 0.03)
Max features: 35, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 130, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 40, num estimators: 100, accuracy: 0.34 (+/- 0.02)
Max features: 40, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max features: 40, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 40, num estimators: 100, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 130, accuracy: 0.45 (+/- 0.04)
Max features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 170, accuracy: 0.45 (+/- 0.04)
Max features: 40, num estimators: 190, accuracy: 0.45 (+/- 0.04)
Max features: 45, num estimators: 100, accuracy: 0.34 (+/- 0.01)
Max features: 45, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 100, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.04)
Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.45 (+/- 0.04)
Max features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 210, accuracy: 0.46 (+/- 0.04)

```

Gambar 4.30 Hasil Soal 8.

4.3.3 Penanganan Error

1. ScreenShoot Error

```

FileNotFoundException: [Error 2] File b'F://Semester 6/Artificial Intelligence/Tugas
4/src/fenny.csv" does not exist! b'F://Semester 6/Artificial Intelligence/Tugas 4/
src/fenny.csv'

```

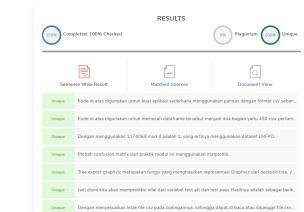
Gambar 4.31 SyntaxError

2. Cara Penanganan Error

- SyntaxError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.3.4 Bukti Tidak Plagiat

**Gambar 4.32** Bukti Tidak Melakukan Plagiat Chapter 4

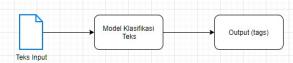
4.3.5 Link Youtube

4.4 1174066 - D.Irga B. Naufal Fakhri

4.4.1 Teori

4.4.1.1 Klasifikasi Teks

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.33 Klasifikasi Teks.

4.4.1.2 Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning

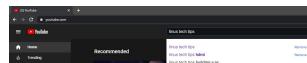
Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut:



Gambar 4.34 Klasifikasi Bunga.

4.4.1.3 Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan

Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut:



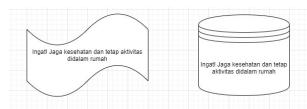
Gambar 4.35 Klasifikasi Teks pada Youtube.

4.4.1.4 Vektorisasi Data

Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

4.4.1.5 Bag of Words

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut:



Gambar 4.36 Bag of Words.

4.4.1.6 TF-IDF

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.37 TF-IDF.

4.4.2 Praktek Program

4.4.2.1 Nomor 1

```

1 import pandas as pd #digunakan untuk mengimport library pandas
    dengan alias pd
2 pd = pd.read_csv("N:/Tugas/Kuliah/Semester 6/Kecerdasan Buatan/
    KB3C Ngerjain/src/1174066/4/csv.csv") #membaca file csv
  
```

```
In [4]: import pandas as pd #digunakan untuk mengimport library pandas dengan alias pd
```

Gambar 4.38 Nomor 1

4.4.2.2 Nomor 2

```
1 d_train=pd[:450] #membagi data training menjadi 450
2 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data yang tersedia
```



Gambar 4.39 Nomor 2

4.4.2.3 Nomor 3

```
1 import pandas as pd #digunakan untuk mengimport library pandas dengan alias pd
2 d = pd.read_csv("N:/Tugas/Kuliah/Semester 6/Kecerdasan Buatan/KB3C Ngerjain/src/1174066/4/Youtube04-Eminem.csv") #Membaca file csv
3
4 from sklearn.feature_extraction.text import CountVectorizer # import fungsi countvectorize dari sklearn
5 vectorizer = CountVectorizer() #membuat instansi CountVectorizer
6
7 dvec = vectorizer.fit_transform(d['CONTENT']) #Memasukkan data ke dvec
8 dvec #Melihat data yang dimasukkan ke dvec
9
10 daptarkata = vectorizer.get_feature_names() #Mendapatkan data dan memasukkannya ke daptarkata
11
12 dshuf = d.sample(frac=1) #Memasukkan sample kedalam variable dshuf
13
14 d_train = dshuf[:300] #Membuat data training
15 d_test = dshuf[300:] #Membuat data test
16
17 d_train_att = vectorizer.fit_transform(d_train['CONTENT']) # Memasukkan data training dari vectorizer
18 d_train_att #Melihat data training
19
20 d_test_att = vectorizer.transform(d_test['CONTENT']) #Memasukkan data test dari vectorizer
21 d_test_att #Melihat data training
```

```

22
23 d_train_label = d_train['CLASS'] #Memberi label
24 d_test_label = d_test['CLASS'] #Memberi Label

```

The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with options like File, Edit, Insert, Cell, Kernel, Help, and a search bar. Below the toolbar, the code cell contains the following Python code:

```

In [42]: import pandas as pd
        df = pd.read_csv('dataset.csv')
        X = df.drop(['label'], axis=1)
        y = df['label'].values.reshape(-1, 1)
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
        vectorizer = CountVectorizer()
        X_train_att = vectorizer.fit_transform(X_train)
        X_test_att = vectorizer.transform(X_test)
        clf = SVC()
        clf.fit(X_train_att, y_train)
        y_train_label = clf.predict(X_train)
        y_test_label = clf.predict(X_test)

```

The code imports pandas, reads a CSV file named 'dataset.csv', drops the 'label' column, and splits the data into training and testing sets using a 70/30 split. It then applies a CountVectorizer to the features and fits an SVC classifier to the training data. Finally, it uses the classifier to predict labels for both training and testing sets.

Gambar 4.40 Nomor 3

4.4.2.4 Nomor 4

```

1 from sklearn import svm #Mengimport svm dari sklearn
2 clfsvm = svm.SVC() #Membuat svc kedalam variable svm
3 clfsvm.fit(d_train_att, d_train_label) #Memprediksi data dari
   data training
4 clfsvm.score(d_test_att, d_test_label) #Memunculkan clf sebagai
   testing yang sudah di training tadi

```

The screenshot shows the execution of the SVM code. The output cell displays the following message:

```

In [43]: clfsvm.score(d_test_att, d_test_label)
Out[43]: 0.9594594594594594

```

This indicates that the classifier achieved a score of approximately 0.959 on the test data.

Gambar 4.41 Nomor 4

4.4.2.5 Nomor 5

```

1 from sklearn import tree #Mengimport tree dari sklearn
2 clftree = tree.DecisionTreeClassifier() #Membuat decision tree
3 clftree.fit(d_train_att, d_train_label) #Memprediksi data dari
   data training
4 clftree.score(d_test_att, d_test_label) #Memunculkan clf sebagai
   testing yang sudah di training tadi

```

The screenshot shows the execution of the Decision Tree code. The output cell displays the following message:

```

In [45]: from sklearn import tree
        clftree = tree.DecisionTreeClassifier()
        clftree.fit(d_train_att, d_train_label)
        clftree.score(d_test_att, d_test_label)
Out[45]: 0.9594594594594594

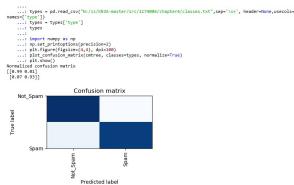
```

This indicates that the decision tree classifier achieved a score of approximately 0.959 on the test data.

Gambar 4.42 Nomor 5

4.4.2.6 Nomor 6

```
1 from sklearn.metrics import confusion_matrix
2 pred_labelstree = clftree.predict(d_test_att)
3 cmtree = confusion_matrix(d_test_label, pred_labelstree)
4 cmtree
5
6 import matplotlib.pyplot as plt
7 import itertools
8 def plot_confusion_matrix(cm, classes,
9                           normalize=False,
10                           title='Confusion matrix',
11                           cmap=plt.cm.Blues):
12     """
13     This function prints and plots the confusion matrix.
14     Normalization can be applied by setting 'normalize=True'.
15     """
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
18         print("Normalized confusion matrix")
19     else:
20         print('Confusion matrix, without normalization')
21
22     print(cm)
23
24     plt.imshow(cm, interpolation='nearest', cmap=cmap)
25     plt.title(title)
26     #plt.colorbar()
27     tick_marks = np.arange(len(classes))
28     plt.xticks(tick_marks, classes, rotation=90)
29     plt.yticks(tick_marks, classes)
30
31     fmt = '.2f' if normalize else 'd'
32     thresh = cm.max() / 2.
33     #for i, j in itertools.product(range(cm.shape[0]), range(cm.
34     #shape[1])):
35     #    plt.text(j, i, format(cm[i, j], fmt),
36     #              horizontalalignment="center",
37     #              color="white" if cm[i, j] > thresh else "black"
38     #)
39
40     plt.tight_layout()
41     plt.ylabel('True label')
42     plt.xlabel('Predicted label')
43 types = pd.read_csv("N:/zz/KB3A-master/src/1174006/chapter4/
44   classes.txt", sep='\s+', header=None, usecols=[1], names=['type
45   '])
46 types = types['type']
47 types
48
49 import numpy as np
50 np.set_printoptions(precision=2)
51 plt.figure(figsize=(4,4), dpi=100)
52 plot_confusion_matrix(cmtree, classes=types, normalize=True)
53 plt.show()
```



Gambar 4.43 Nomor 6

4.4.2.7 Nomor 7

```

1 from sklearn.model_selection import cross_val_score
2
3 scorestree = cross_val_score(clftree, d_train_att, d_train_label,
4                             cv=5)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
6                                           scorestree.std() * 2))
7
8 scoresvm = cross_val_score(clfsvm, d_train_att, d_train_label,
9                             cv=5)
10 print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(),
11                                         scoresvm.std() * 2))
12
13 scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
14 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
15                                         () * 2))
    
```

```

In [8]: from sklearn.model_selection import cross_val_score
... scorestree = cross_val_score(clftree, d_train_att, d_train_label, cv=5)
... print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
...                                         scorestree.std() * 2))
... scoresvm = cross_val_score(clfsvm, d_train_att, d_train_label, cv=5)
... print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(),
...                                         scoresvm.std() * 2))
... scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
... print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
...                                         () * 2))

Accuracy: 0.97 (+/- 0.00)
Accuracy: 0.99 (+/- 0.00)
Accuracy: 0.99 (+/- 0.00)

```

Gambar 4.44 Nomor 7

4.4.2.8 Nomor 8

```

1 max_features_opts = range(5, 50, 5)
2 n_estimators_opts = range(10, 200, 20)
3 rf_params = np.empty((len(max_features_opts)*len(
4     n_estimators_opts), 4), float)
5 i = 0
6 for max_features in max_features_opts:
7     for n_estimators in n_estimators_opts:
8         clf = RandomForestClassifier(max_features=max_features,
9             n_estimators=n_estimators)
10        scores = cross_val_score(clf, d_train_att, d_train_label,
11                                  cv=5)
12        rf_params[i, 0] = max_features
13        rf_params[i, 1] = n_estimators
14        rf_params[i, 2] = scores.mean()
15        rf_params[i, 3] = scores.std() * 2
16 i += 1
    
```

```

10     rf_params[i,1] = n_estimators
11     rf_params[i,2] = scores.mean()
12     rf_params[i,3] = scores.std() * 2
13     i += 1
14     print("Max features: %d, num estimators: %d, accuracy: "
15           "%0.2f (+/- %0.2f)" % (max_features, n_estimators,
16           scores.mean(), scores.std() * 2))

```

```

In [20]: max_features+=np.array([0,1,2])
...:
...: #print(rf_params)
...: for i in range(1,10):
...:     print("Max features: %d, num estimators: %d, accuracy: %0.2f (+/- %0.2f)" % (max_features, n_estimators,
...:         scores.mean(), scores.std() * 2))

Max Features: 0, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 1, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 2, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 3, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 4, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 5, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 6, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 7, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 8, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 9, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 10, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 11, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 12, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 13, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 14, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 15, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 16, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 17, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 18, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 19, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 20, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 21, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 22, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 23, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 24, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 25, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 26, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 27, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 28, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 29, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 30, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 31, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 32, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 33, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 34, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 35, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 36, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 37, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 38, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 39, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 40, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 41, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 42, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 43, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 44, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 45, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 46, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 47, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 48, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 49, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 50, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 51, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 52, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 53, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 54, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 55, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 56, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 57, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 58, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 59, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 60, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 61, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 62, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 63, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 64, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 65, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 66, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 67, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 68, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 69, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 70, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 71, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 72, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 73, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 74, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 75, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 76, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 77, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 78, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 79, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 80, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 81, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 82, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 83, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 84, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 85, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 86, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 87, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 88, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 89, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 90, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 91, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 92, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 93, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 94, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 95, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 96, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 97, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 98, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 99, num estimators: 10, accuracy: 0.80 (+/- 0.00)
Max Features: 100, num estimators: 10, accuracy: 0.80 (+/- 0.00)

```

Gambar 4.45 Nomor 8

4.4.3 Penanganan Error

1. FileNotFoundError

```

File "pandas\_libs\parsers.pyx", line 689, in
pandas\_libs.parsers.TextReader._setup_parser_source
FileNotFoundError: [Errno 2] No such file or directory: 'C:/Users/Bustan/KSIC/Ngerjain/src/1174006/A.csv'

```

Gambar 4.46 FileNotFoundError

2. Cara Penanganan Error

- FileNotFoundError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.4.4 Bukti Tidak Plagiat



Gambar 4.47 Bukti Tidak Melakukan Plagiat Chapter 4

4.4.5 Link Youtube

<https://youtu.be/Lw0r-UAb8jY>

4.5 1174083 - Bakti Qilan Mufid

Chapter-4 Klasifikasi Teks

4.5.1 Teori

4.5.1.1 Jelaskan apa itu klasifikasi teks, sertakan gambar iustrasi buatan sendiri

Klasifikasi teks merupakan salah satu tugas terpenting dalam Pemrosesan Bahasa Alami (Natural Language Processing). Ini adalah proses mengklasifikasikan string teks atau dokumen ke dalam kategori yang berbeda, tergantung pada konten string. Klasifikasi teks memiliki berbagai aplikasi, seperti mendeteksi sentimen pengguna dari tweet, mengklasifikasikan email sebagai spam atau ham, mengklasifikasikan posting blog ke dalam kategori yang berbeda, penandaan otomatis permintaan pelanggan, dan sebagainya. Berikut adalah contoh dari Klasifikasi Teks. Contohnya, misal kita ingin mencari kata dog, is, table, on, the . kemudian jika kata yang dimaksud sesuai maka akan menampilkan bilangan biner 1 dan jika salah 0. Seperti dibawah ini :

the dog is on the table



Gambar 4.48 Klasifikasi Teks

4.5.1.2 Jelaskan mengapa klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri

Dikarenakan tidak semua bunga memiliki ciri - ciri yang sama. Atau dalam kata lain terdapat data noise dalam klasifikasi bunga sehingga tidak bisa menggunakan machine learning. Contohnya Anggrek memiliki warna ungu, dengan jumlah kelopak 5. Kemudian ada bunga warna ungu dengan jumlah kelopak yang sama namun ternyata bukan anggrek dan kategorinya banyak sekali. Bahkan ada bunga yang tidak jelas apakah warnanya sesuai atau tidak, sehingga bisa menyebabkan data noise.



Gambar 4.49 Klasifikasi Bunga

4.5.1.3 *Jelaskan bagaimana teknik pembelajaran mesin pada teks pada kata-kata yang digunakan di youtube, jelaskan arti per atribut data csv dan sertakan ilustrasi buatan sendiri.*

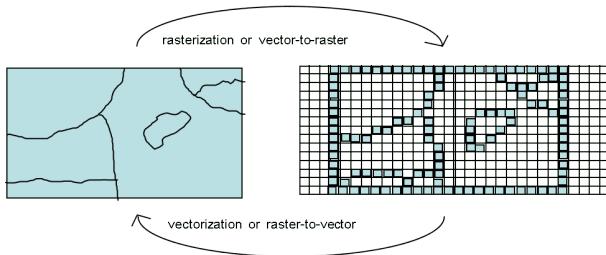
Menggunakan teknik bag-of-words pada klasifikasi berbasis text dan kata untuk mengklasifikasikan komentar yang ada di internet sebagai spam atau bukan. Misalkan pada kolom komentar dapat di cek seberapa sering suatu kata muncul dalam kalimat. Setiap kata dapat dijadikan baris dan kolomnya ini merupakan kategori kata terbut, apakah masuk kedalam spam atau tidak. dan contoh lainnya yaitu pada Caption. dimana akan muncul subtitle secara otomatis dari youtube menggunakan sensor suara yang disesuaikan dengan kata yang telah ditentukan. Contohnya seperti berikut :

| CONTENT | CLASS |
|--|-------|
| Huh, anyway check out this you[tube] channel: kobyoshi02 | 1 |
| Hey guys check out my new channel and our first vid THIS IS US THE MONKEYS!!! I'm the monkey in the white shirt,please leave a like comment and please subscribe!!!! | 1 |
| just for test I have to say murdev.com | 1 |
| me shaking my sexy ass on my channel enjoy ^_^ | 1 |
| watch?v=vtaRGgvGtW0 Check this out . | 1 |
| Hey, check out my new website!! This site is about kids stuff. kidsmediausa . com | 1 |
| Subscribe to my channel | 1 |
| i turned it on mute as soon as i came on i just wanted to check the views... | 0 |
| You should check my channel for Funny VIDEOS!! | 1 |
| and u shoud d check my channel and tell me what I should do next! | 1 |

Gambar 4.50 Klasifikasi Spam Comment di Youtube

4.5.1.4 *Jelaskan apa yang dimaksud vektorisasi data.*

Vektorisasi adalah proses konversi data raster(gambar, pindaian) menjadi data vektor yang lebih umum disebut dengan istilah digitalisasi adapun aktifitasnya disebut digitasi. Wujud digitalisasi ini diklasifikasikan secara spesifik dalam tema-tema tertentu yang direpresentasikan oleh bentuk garis, poligon dan titik. Pada akhirnya proses vektorisasi ini menghasilkan suatu wujud topografi yang menggambarkan keadaan permukaan bumi atau bentang alam. Sifat data yang geometris menunjukkan ukuran dimensi yang sesungguhnya.



Gambar 4.51 Vektorisasi dan Rasterisasi

4.5.1.5 *Jelaskan apa itu bag of words dengan kata-kata yang sederhana dan ilustrasi sendiri.*

bag-of-words merupakan representasi teks yang menggambarkan kemunculan kata-kata dalam dokumen. Pengelompokan kata kata kedalam perhitungan, berapakah sebuah kata muncul dalam satu kalimat. Disebut "bag" kata-kata, karena informasi tentang susunan atau struktur kata dalam dokumen dibuang. Model ini hanya berkaitan dengan apakah kata-kata yang diketahui muncul dalam dokumen, bukan di mana dalam dokumen. Contohnya disini akan melihat kemunculan kata dari kalimat :

1. I Love Dogs
2. I hate dogs and knitting
3. Knitting is my hobby and passion.

| | I | love | dogs | hate | and | knitting | is | my | hobby | passion |
|-------|---|------|------|------|-----|----------|----|----|-------|---------|
| Doc 1 | 1 | 1 | 1 | | | | | | | |
| Doc 2 | 1 | | 1 | 1 | 1 | 1 | | | | |
| Doc 3 | | | | | 1 | 1 | 1 | 2 | 1 | 1 |

Gambar 4.52 Bag-Of-Words

4.5.1.6 *Jelaskan apa itu TF-IDF, ilustrasikan dengan gambar sendiri.*

TF-IDF memberi kita frekuensi kata dalam setiap dokumen dalam korpus atau mengganti data jadi number. Ini adalah rasio berapa kali kata itu muncul dalam dokumen dibandingkan dengan jumlah total kata dalam dokumen itu. Itu meningkat seiring jumlah kemunculan kata itu di dalam dokumen meningkat. Setiap dokumen memiliki tf sendiri. Dalam ilustrasi disini saya akan mengganti contoh Bag of Words menjadi bentuk TF-IDF.

| | | | | | | | | | | |
|-------|------|-------------|------|-------------|------|----------|-------------|-------------|-------------|-------------|
| | I | love | dogs | hate | and | knitting | is | my | hobby | passion |
| Doc 1 | 0.18 | 0.48 | 0.18 | | | | | | | |
| Doc 2 | 0.18 | | 0.18 | 0.48 | 0.18 | 0.18 | | | | |
| Doc 3 | | | | | 0.18 | 0.18 | 0.48 | 0.95 | 0.48 | 0.48 |

Gambar 4.53 Contoh TF-IDF

4.5.2 Praktek

4.5.2.1 buat aplikasi klasifikasi sederhana menggunakan pandas, buat data dummy format csv sebanyak 500 baris dan melakukan load ke dataframe panda.jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```
1 import pandas as pd #import package pandas, lalu dialiaskan
    menjadi pd.
2 marvel = pd.read_csv ('E:/backup/sem 6/Kecerdasan Buatan/KB3C –
    Copy/src/1174083/src4/Marvel.csv', sep=',') #membaca file csv
    dimana data pada file csv dipisahkan oleh koma, lalu
    ditampung di variable marvel.
```

Listing 4.1 kodingan praktek no. 1

| Index | name | ID | ALIGN | EYE | HAIR |
|-------|---|------------|---------|---------------|---------------|
| 400 | (Earth-616) Identity | Characters | Neutral | green eyes | black hair |
| 487 | Keniuchio Public | Characters | Neutral | Brown Eyes | Black Hair |
| 488 | Harada (Earth-616) Identity | Characters | Neutral | White Eyes | Silver Hair |
| 489 | Irene Adler Secret | Characters | Good | Black Eyes | Green Hair |
| 490 | Marrina Secret | Characters | Good | Blue Eyes | White Hair |
| 491 | Smallwood (Earth-616) Identity | Characters | Bad | Green Eyes | Auburn Hair |
| 492 | Fred Davis Secret Jr. (Earth-616) Identity | Characters | Neutral | Variable Eyes | Variable Hair |
| 493 | Zelda DuBois Public (Earth-616) Identity | Characters | Good | nan | nan |
| 494 | Beyonder Secret (Earth-616) Identity | Characters | Bad | Brown Eyes | Black Hair |
| 495 | Roxanne Washington (...) Identity | Characters | Neutral | Yellow Eyes | Red Hair |
| 496 | Bonita Juarez Secret (Earth-616) Identity | Characters | Good | White Eyes | White Hair |
| 497 | Surtur No Dual (Earth-616) Identity | Characters | Good | nan | Brown Hair |
| 498 | Stranger Secret (Cosmic Being-616) Identity | Characters | Bad | Hazel Eyes | Brown Hair |
| 499 | Brandy Clark Public (Earth-616) Identity | Characters | Neutral | nan | White Hair |
| | Lillian Crawley (Earth-616) Identity | Characters | Good | | |
| | Ichabod Secret (Earth-616) Identity | Characters | Good | | |

Gambar 4.54 hasil praktek soal no. 1

4.5.2.2 dari dataframe tersebut dipecah menjadi dua dataframe yaitu 450 row pertama dan 50 row sisanya(harus beda dengan teman sekelas)

```

1 marvel1 , marvel2 = marvel[:450] , marvel[450:] #membagi data
  menjadi dua bagian , variable marvel1 untuk menampung 450
  baris data pertama , variable marvel2 untuk menampung 50 baris
  data terakhir .

```

Listing 4.2 kodingan praktek no. 2

| Name | Type | Size | Value |
|---------|-----------|----------|--|
| marvel | DataFrame | (500, 7) | Column names: name, ID, ALIGN, EYE, HAIR, SEX, ALIVE |
| marvel1 | DataFrame | (450, 7) | Column names: name, ID, ALIGN, EYE, HAIR, SEX, ALIVE |
| marvel2 | DataFrame | (50, 7) | Column names: name, ID, ALIGN, EYE, HAIR, SEX, ALIVE |

Gambar 4.55 hasil praktek soal no. 2

4.5.2.3 praktekkan vektorisasi dan klasifikasi dari data(NPM mod 4, jika 0 maka ketty perry, 1 LMFAO, 2 Eminem, 3 Shakira) dengan Decission Tree. Tunjukkan keluaranya dari komputer sendiri dan artikan maksud luaran yang di dapatkan

```

1 print(1174083%4) #hasilnya 3 , maka Shakira

```

Listing 4.3 1174083 mod 4

```

1 # Vektorisasi Data
2 import pandas as pd
3 d = pd.read_csv("Youtube05-Shakira.csv")
4
5 from sklearn.feature_extraction.text import CountVectorizer
6 vectorizer = CountVectorizer()
7
8 dvec = vectorizer.fit_transform(d['CONTENT'])
9 dvec
10
11 daptarkata = vectorizer.get_feature_names()
12
13 dshuf = d.sample(frac=1)
14
15 d_train = dshuf[:300]
16 d_test = dshuf[300:]
17
18 d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
19 d_train_att
20
21 d_test_att = vectorizer.transform(d_test['CONTENT'])
22 d_test_att
23
24 d_train_label = d_train['CLASS']
25 d_test_label = d_test['CLASS']

```

Listing 4.4 kodingan praktek no. 3

```
In [7]: import pandas as pd
...: d = pd.read_csv("Youtube05-Shakira.csv")
...:
...: from sklearn.feature_extraction.text import CountVectorizer
...: vectorizer = CountVectorizer()
...:
...: dvec = vectorizer.fit_transform(d['CONTENT'])
...: dvec
...:
...: daptarkata = vectorizer.get_feature_names()
...:
...: dshuf = d.sample(frac=1)
...:
...: d_train = dshuf[:300]
...: d_test = dshuf[300:]
...:
...: d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
...: d_train_att
...:
...: d_test_att = vectorizer.transform(d_test['CONTENT'])
...: d_test_att
...:
...: d_train_label = d_train['CLASS']
...: d_test_label = d_test['CLASS']
```

Gambar 4.56 hasil praktek soal no. 3(1)

Vektorisasi data content dari file Youtube04_Shakira.CSV

| Name | Type | Size | Value |
|---------------|-----------|----------|---|
| d | DataFrame | (370, 5) | Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS |
| d_test | DataFrame | (70, 5) | Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS |
| d_test_label | Series | (70,) | Series object of pandas.core.series module |
| d_train | DataFrame | (300, 5) | Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS |
| d_train_label | Series | (300,) | Series object of pandas.core.series module |
| daptarkata | list | 1357 | ['00', '000', '0687119038', '08', '10', '100', '101721377578919894134' ...] |
| dshuf | DataFrame | (370, 5) | Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS |

Gambar 4.57 hasil praktek soal no. 3(2)

4.5.2.4 Cobalah klarifikasi dari data vektorisasi yang di tentukan di nomor sebelumnya dengan klasifikasi SVM. Tunjukkan keluaranya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

```
1 from sklearn import svm
2 clfsvm = svm.SVC()
3 clfsvm.fit(d_train_att, d_train_label)
4 clfsvm.score(d_test_att, d_test_label)
```

Listing 4.5 kodingan praktek no. 4

```
In [9]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The
default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
Out[9]: 0.6857142857142857
```

Gambar 4.58 hasil praktek soal no. 4

4.5.2.5 Cobalah klasifikasikan dari data vektorisasi yang ditentukan dari nomor sebelumnya dengan klasifikasi Decision Tree. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

```
1 from sklearn import tree
2 clftree = tree.DecisionTreeClassifier()
3 clftree.fit(d_train_att, d_train_label)
4 clftree.score(d_test_att, d_test_label)
```

Listing 4.6 kodingan praktek no. 5

```
In [8]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
...: clftree.score(d_test_att, d_test_label)
Out[8]: 0.9285714285714286
```

Gambar 4.59 hasil praktek soal no. 5

4.5.2.6 Plotlah confusion matrix dari praktek modul ini menggunakan matplotlib. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

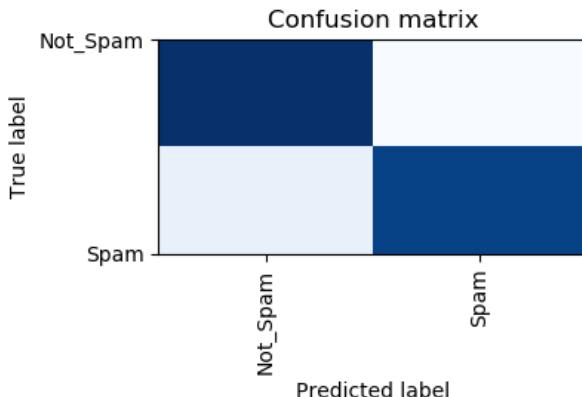
```
1 from sklearn.metrics import confusion_matrix
2 pred_labeltree = clftree.predict(d_test_att)
3 cmmtree = confusion_matrix(d_test_label, pred_labeltree)
4 cmmtree
5
6 import matplotlib.pyplot as plt
7 import itertools
8 def plot_confusion_matrix(cm, classes,
9                         normalize=False,
10                         title='Confusion matrix',
11                         cmap=plt.cm.Blues):
12     """
13     This function prints and plots the confusion matrix.
14     Normalization can be applied by setting `normalize=True`.
15     """
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
18         print("Normalized confusion matrix")
19     else:
```

```

20     print('Confusion matrix, without normalization')
21
22     print(cm)
23
24     plt.imshow(cm, interpolation='nearest', cmap=cmap)
25     plt.title(title)
26     #plt.colorbar()
27     tick_marks = np.arange(len(classes))
28     plt.xticks(tick_marks, classes, rotation=90)
29     plt.yticks(tick_marks, classes)
30
31     fmt = '.2f' if normalize else 'd'
32     thresh = cm.max() / 2.
33
34     plt.tight_layout()
35     plt.ylabel('True label')
36     plt.xlabel('Predicted label')
37
38 types = pd.read_csv("classes.txt", sep='\s+', header=None, usecols=[1], names=['type'])
39 types = types['type']
40 types
41
42 import numpy as np
43 np.set_printoptions(precision=2)
44 plt.figure(figsize=(4,4), dpi=100)
45 plot_confusion_matrix(cmtree, classes=types, normalize=True)
46 plt.show()

```

Listing 4.7 kodingan praktek no. 6(1)



Gambar 4.60 hasil praktek soal no. 6(1)

Plot confusion matrix dari klasifikasi Decission Tree.

```

1 from sklearn.metrics import confusion_matrix
2 pred_labelssvm = clfsvm.predict(d_test_att)

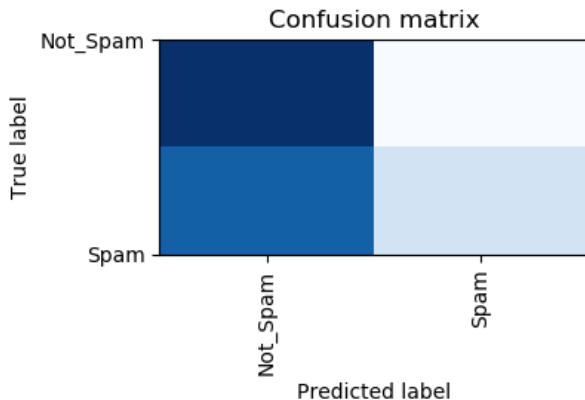
```

```

3 cmsvm = confusion_matrix(d_test_label , pred_labelssvm)
4 cmsvm
5
6 import matplotlib.pyplot as plt
7 import itertools
8 def plot_confusion_matrix(cm, classes,
9                           normalize=False,
10                           title='Confusion matrix',
11                           cmap=plt.cm.Blues):
12     """
13     This function prints and plots the confusion matrix.
14     Normalization can be applied by setting 'normalize=True'.
15     """
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
18         print("Normalized confusion matrix")
19     else:
20         print('Confusion matrix, without normalization')
21
22     print(cm)
23
24     plt.imshow(cm, interpolation='nearest', cmap=cmap)
25     plt.title(title)
26     #plt.colorbar()
27     tick_marks = np.arange(len(classes))
28     plt.xticks(tick_marks, classes, rotation=90)
29     plt.yticks(tick_marks, classes)
30
31     fmt = '.2f' if normalize else 'd'
32     thresh = cm.max() / 2.
33     #for i, j in itertools.product(range(cm.shape[0]), range(cm.
34     #shape[1])):
35     #    plt.text(j, i, format(cm[i, j], fmt),
36     #              horizontalalignment="center",
37     #              color="white" if cm[i, j] > thresh else "black"
38     )
39
40     plt.tight_layout()
41     plt.ylabel('True label')
42     plt.xlabel('Predicted label')
43
44 types = pd.read_csv("classes.txt",sep='\s+', header=None,usecols
45 = [1], names=['type'])
46 types = types['type']
47
48 import numpy as np
49 np.set_printoptions(precision=2)
50 plt.figure(figsize=(4,4), dpi=100)
51 plot_confusion_matrix(cmsvm, classes=types, normalize=True)
52 plt.show()

```

Listing 4.8 kodingan praktik no. 6(2)



Gambar 4.61 hasil praktek soal no. 6(2)

Plot confusion matrix dari klasifikasi SVM.

```

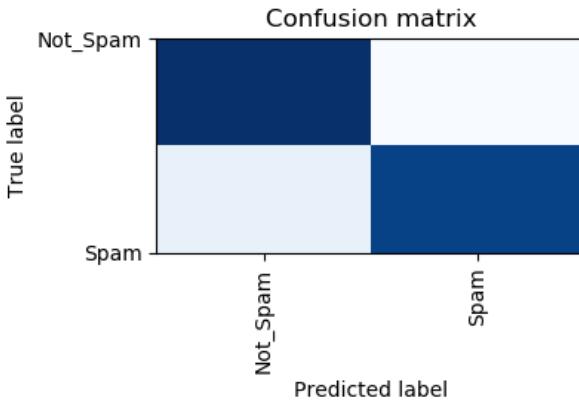
1 from sklearn.metrics import confusion_matrix
2 pred_labels = clf.predict(d_test_att)
3 cm = confusion_matrix(d_test_label, pred_labels)
4
5 import matplotlib.pyplot as plt
6 import itertools
7 def plot_confusion_matrix(cm, classes,
8                           normalize=False,
9                           title='Confusion matrix',
10                          cmap=plt.cm.Blues):
11     """
12     This function prints and plots the confusion matrix.
13     Normalization can be applied by setting 'normalize=True'.
14     """
15     if normalize:
16         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
17         print("Normalized confusion matrix")
18     else:
19         print('Confusion matrix, without normalization')
20
21     print(cm)
22
23     plt.imshow(cm, interpolation='nearest', cmap=cmap)
24     plt.title(title)
25     #plt.colorbar()
26     tick_marks = np.arange(len(classes))
27     plt.xticks(tick_marks, classes, rotation=90)
28     plt.yticks(tick_marks, classes)
29
30     fmt = '.2f' if normalize else 'd'
31     thresh = cm.max() / 2.
32     #for i, j in itertools.product(range(cm.shape[0]), range(cm.
33     #shape[1])):
34     #    plt.text(j, i, format(cm[i, j], fmt),
35

```

```

34     #           horizontalalignment="center",
35     #           color="white" if cm[i, j] > thresh else "black
36     ")
37     plt.tight_layout()
38     plt.ylabel('True label')
39     plt.xlabel('Predicted label')
40
41 types = pd.read_csv("classes.txt", sep='\s+', header=None, usecols
42     =[1], names=['type'])
43 types = types['type']
44 types
45 import numpy as np
46 np.set_printoptions(precision=2)
47 plt.figure(figsize=(4,4), dpi=100)
48 plot_confusion_matrix(cmtree, classes=types, normalize=True)
49 plt.show()

```

Listing 4.9 kodingan praktek no. 6(3)**Gambar 4.62** hasil praktek soal no. 6(3)

Plot confusion matrix dari klasifikasi Random Forest.

4.5.2.7 jalankan program cross validaiton pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```

1 from sklearn.model_selection import cross_val_score
2
3 scorestree = cross_val_score(clftree, d_train_att, d_train_label,
4     cv=5)
4 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
5     scorestree.std() * 2))

```

```

5
6 scoreSVM = cross_val_score(clfSVM, d_train_att, d_train_label,
7     cv=5)
7 print("Accuracy: %0.2f (+/- %0.2f)" % (scoreSVM.mean(),
8     scoreSVM.std() * 2))
8
9 scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
10 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
10     () * 2))

```

Listing 4.10 kodingan praktek no. 7

Accuracy: 0.92 (+/- 0.05)
Accuracy: 0.66 (+/- 0.05)
Accuracy: 0.94 (+/- 0.03)

Gambar 4.63 hasil praktek soal no. 7

4.5.2.8 Buatlah program pengamatan komponen informasi pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```

1 max_features_opts = range(5, 50, 5)
2 n_estimators_opts = range(10, 200, 20)
3 rf_params = np.empty((len(max_features_opts)*len(
    n_estimators_opts),4), float)
4 i = 0
5 for max_features in max_features_opts:
6     for n_estimators in n_estimators_opts:
7         clf = RandomForestClassifier(max_features=max_features,
8             n_estimators=n_estimators)
9         scores = cross_val_score(clf, d_train_att, d_train_label,
10             cv=5)
11         rf_params[i,0] = max_features
12         rf_params[i,1] = n_estimators
13         rf_params[i,2] = scores.mean()
14         rf_params[i,3] = scores.std() * 2
15         i += 1
16     print("Max features: %d, num estimators: %d, accuracy:
16         %0.2f (+/- %0.2f)" % (max_features,
16             n_estimators, scores.mean(), scores.std() * 2))

```

Listing 4.11 kodingan praktek no. 8

```
Max features: 5, num estimators: 10, accuracy: 0.89 (+/- 0.09)
Max features: 5, num estimators: 30, accuracy: 0.90 (+/- 0.07)
Max features: 5, num estimators: 50, accuracy: 0.91 (+/- 0.08)
Max features: 5, num estimators: 70, accuracy: 0.92 (+/- 0.07)
Max features: 5, num estimators: 90, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 110, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 130, accuracy: 0.94 (+/- 0.04)
Max features: 5, num estimators: 150, accuracy: 0.93 (+/- 0.05)
Max features: 5, num estimators: 170, accuracy: 0.93 (+/- 0.05)
Max features: 5, num estimators: 190, accuracy: 0.92 (+/- 0.06)
Max features: 10, num estimators: 10, accuracy: 0.92 (+/- 0.07)
Max features: 10, num estimators: 30, accuracy: 0.93 (+/- 0.04)
Max features: 10, num estimators: 50, accuracy: 0.94 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.93 (+/- 0.05)
Max features: 10, num estimators: 90, accuracy: 0.93 (+/- 0.04)
Max features: 10, num estimators: 110, accuracy: 0.92 (+/- 0.06)
Max features: 10, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 10, num estimators: 150, accuracy: 0.94 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.93 (+/- 0.04)
Max features: 10, num estimators: 190, accuracy: 0.93 (+/- 0.04)
Max features: 15, num estimators: 10, accuracy: 0.90 (+/- 0.05)
Max features: 15, num estimators: 30, accuracy: 0.93 (+/- 0.04)
Max features: 15, num estimators: 50, accuracy: 0.93 (+/- 0.05)
Max features: 15, num estimators: 70, accuracy: 0.92 (+/- 0.05)
Max features: 15, num estimators: 90, accuracy: 0.94 (+/- 0.04)
Max features: 15, num estimators: 110, accuracy: 0.94 (+/- 0.03)
Max features: 15, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 15, num estimators: 150, accuracy: 0.94 (+/- 0.03)
Max features: 15, num estimators: 170, accuracy: 0.94 (+/- 0.03)
Max features: 15, num estimators: 190, accuracy: 0.93 (+/- 0.04)
Max features: 20, num estimators: 10, accuracy: 0.91 (+/- 0.05)
Max features: 20, num estimators: 30, accuracy: 0.91 (+/- 0.06)
Max features: 20, num estimators: 50, accuracy: 0.94 (+/- 0.04)
Max features: 20, num estimators: 70, accuracy: 0.92 (+/- 0.04)
Max features: 20, num estimators: 90, accuracy: 0.93 (+/- 0.04)
Max features: 20, num estimators: 110, accuracy: 0.93 (+/- 0.04)
Max features: 20, num estimators: 130, accuracy: 0.94 (+/- 0.03)
Max features: 20, num estimators: 150, accuracy: 0.93 (+/- 0.04)
Max features: 20, num estimators: 170, accuracy: 0.93 (+/- 0.04)
```

Gambar 4.64 hasil praktek soal no. 8(1)

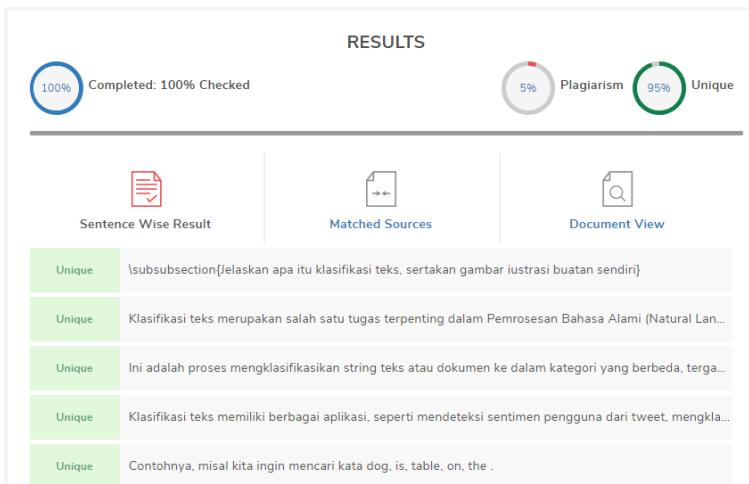
```
Max features: 25, num estimators: 150, accuracy: 0.93 (+/- 0.04)
Max features: 25, num estimators: 170, accuracy: 0.94 (+/- 0.03)
Max features: 25, num estimators: 190, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 10, accuracy: 0.93 (+/- 0.06)
Max features: 30, num estimators: 30, accuracy: 0.93 (+/- 0.05)
Max features: 30, num estimators: 50, accuracy: 0.95 (+/- 0.04)
Max features: 30, num estimators: 70, accuracy: 0.93 (+/- 0.04)
Max features: 30, num estimators: 90, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 110, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 130, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 150, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 170, accuracy: 0.93 (+/- 0.04)
Max features: 30, num estimators: 190, accuracy: 0.93 (+/- 0.04)
Max features: 35, num estimators: 10, accuracy: 0.91 (+/- 0.05)
Max features: 35, num estimators: 30, accuracy: 0.94 (+/- 0.05)
Max features: 35, num estimators: 50, accuracy: 0.93 (+/- 0.04)
Max features: 35, num estimators: 70, accuracy: 0.94 (+/- 0.03)
Max features: 35, num estimators: 90, accuracy: 0.94 (+/- 0.03)
Max features: 35, num estimators: 110, accuracy: 0.93 (+/- 0.04)
Max features: 35, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 35, num estimators: 150, accuracy: 0.94 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.93 (+/- 0.04)
Max features: 35, num estimators: 190, accuracy: 0.93 (+/- 0.04)
Max features: 40, num estimators: 10, accuracy: 0.91 (+/- 0.06)
Max features: 40, num estimators: 30, accuracy: 0.93 (+/- 0.04)
Max features: 40, num estimators: 50, accuracy: 0.94 (+/- 0.03)
Max features: 40, num estimators: 70, accuracy: 0.94 (+/- 0.03)
Max features: 40, num estimators: 90, accuracy: 0.94 (+/- 0.03)
Max features: 40, num estimators: 110, accuracy: 0.93 (+/- 0.05)
Max features: 40, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 40, num estimators: 150, accuracy: 0.93 (+/- 0.05)
Max features: 40, num estimators: 170, accuracy: 0.93 (+/- 0.04)
Max features: 40, num estimators: 190, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 10, accuracy: 0.91 (+/- 0.06)
Max features: 45, num estimators: 30, accuracy: 0.94 (+/- 0.04)
Max features: 45, num estimators: 50, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 70, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 90, accuracy: 0.94 (+/- 0.03)
Max features: 45, num estimators: 110, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 150, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 170, accuracy: 0.94 (+/- 0.03)
Max features: 45, num estimators: 190, accuracy: 0.93 (+/- 0.04)
```

Gambar 4.65 hasil praktek soal no. 8(2)

4.5.3 Penanganan Error

pada chapter 4 ini saya tidak menemukan error.

4.5.4 Bukti Tidak Plagiat



Gambar 4.66 Bukti tidak plagiat

4.5.5 Link Youtube

<https://youtu.be/la8Jptbm3Mo>

4.6 Muhammad Reza Syachrani - 1174084

4.6.1 Teori

1. Jelaskan apa itu klasifikasi teks, dan gambar ilustrasi
klasifikasi teks adalah cara untuk mengklasifikasikan atau mengelompokkan teks berdasarkan parameter tertentu baik itu jenis teks atau jenis dari dokumen yang terdapat kumpulan teks didalamnya.



Gambar 4.67 contoh klasifikasi teks

2. Jelaskan mengapa klasifikasi pada bunga tidak bisa menggunakan machine learning, sertakan ilustrasi.

karna melakukan klasifikasi pada bunga tidak dapat menggunakan mesin learning karena terdapat banyak jenis-jenis bunga yang mirip hingga ada yang sama persis tetapi tidak sama. oleh karena itu klasifikasi bunga tida bisa dilakukan oleh mesin learning dikarenakan jika inputan ciri-ciri tidak sesuai maka bunga di inputkan kemungkinan jawaban dari mesin learning itu tidak tepat contoh menginputkan ciri-ciri bunga serupa tetapi pada mesin learning menjawab itu merupakan ciri-ciri bunga teratai.



Gambar 4.68 contoh klasifikasi bunga

3. Jelaskan bagaimana teknik pembelajaran mesin pada teks pada kata-kata yang digunakan di youtube.

cara pembelajaran teks yang digunakan pada youtube yaitu dengan cara menyimpan inputan teks pada menu pencarian youtube. sehingga pada saat kita akan mencari data yang serupa maka youtube menyediakan rekomendasi-rekomendasi dari pencaharian. contoh pada menu pencarian kita menulis kata sa maka akan muncul banyak rekomendasi yang serupa yang sering di cari oleh orang dalam jangka waktu tertentu.

sa

satu hati sampai mati
samudra cinta
sara wijayanto rumah ruben onsu
sara wijaya
samudra cinta 22 maret 2020
saaih halilintar
sambel trasie happy asmara
samudra cinta 21 maret 2020
sapi
sampek tuwek lirik
satu nama tetap dihati
sarah sheila kamalia
sakit pinggang
sakit dalam bercinta ipank lirik

Laporan prediksi penulisiran

Gambar 4.69 contoh teknik pembelajaran mesin

4. Jelaskan apa yang dimaksud vektorisasi data

vektorisasi data merupakan pembagian data menjadi bagian-bagian yang lebih sederhana, contoh pada satu paragraf terdiri dari 200 kata kemudian dilakukan vektorisasi dengan cara membagi-bagi kata dalam paragraf tersebut ke dalam kalimat-kalimat yang terpisah kemudian dipecah lagi menjadi data dalam perkata selanjutnya kata-kata tersebut dijemahkan.

5. Jelaskan apa itu bag of words dan ilustrasi

bag of words merupakan representasi teks yang menggambarkan kemunculan kata-kata dalam dokumen. ePngelompokan kata-kata kedalam perhitungan, berapakah sebuah kata muncul dalam satu kalimat. Disebut "tas" kata-kata, karena informasi tentang susunan atau struktur kata dalam dokumen dibuang. Model ini hanya berkaitan dengan apakah kata-kata yang diketahui muncul dalam dokumen, bukan di mana dalam dokumen. Contohnya disini akan melihat kemunculan kata dari kalimat :

- Ariq suka menonton film. Alvan juga suka film.

- Alvan juga suka menonton anime.

| | Arig | suka | menonton | film | Alvan | juga | anime |
|------|------|------|----------|------|-------|------|-------|
| Doc1 | 1 | 2 | 1 | 2 | 1 | 1 | |
| Doc2 | | 1 | 1 | | 1 | 1 | 1 |

Gambar 4.70 contoh bag of words

6. Jelaskan apa itu TF-IDF.

TF-IDF memberi frekuensi kata dalam setiap dokumen dalam mengganti data jadi number. Ini adalah rasio berapa kali kata itu muncul dalam dokumen dibandingkan dengan jumlah total kata dalam dokumen. Itu meningkat sejalan dengan jumlah kemunculan kata itu di dalam dokumen meningkat. Setiap dokumen memiliki tf sendiri. rumus TF-IDF :

$$W_{t,d} = TF_{t,d} * IDF_t \quad (1)$$

Keterangan :

$W_{t,d}$ = bobot dari t (*term*) dalam satu dokumen

$TF_{t,d}$ = frekuensi kemunculan t (*term*) dalam dokumen d

IDF_t = *Inverse document frequency*, dimana

$$IDF_t = \log\left(\frac{N}{n_t}\right) \quad (2)$$

Keterangan :

N = jumlah semua dokumen

n_t = jumlah dokumen yang mengandung *term t*

Gambar 4.71 rumus Tf-IDF

4.6.2 Praktek

1. No. 1

```

1 # In [1]:
2 import pandas as pd #mengimport librari padas
3 us = pd.read_csv("D:/Git Kecerdasan Buatan/KB3C/src
        /1174084/4/us-500.csv") # variabel us untuk membaca file
        csv menggunakan fungsi read csv dari padas
4 print(len(us)) #melihat jumlah dari baris data yang telah di
        import
5 print(us.head()) #melihat lima baris pertama data yang telah
        di import
6 print(us.shape) #mengetahui banyak baris dan kolom dari data

```

```

500
   first_name ... web
0    James ... http://www.bentonjohnbjr.com
1 Josephine ... http://www.chanayjeffreyaesq.com
2      Art ... http://www.chemeljameslcpa.com
3     Lenna ... http://www.feltzprintingservice.com
4   Donette ... http://www.printingdimensions.com

[5 rows x 12 columns]
(500, 12)

```

Gambar 4.72 aplikasi sederhana pandas

2. No. 2

```

1 # In [5]:
2 data_training = us[:450] #membuat data training sebanyak 450
  baris
3 data_testing = us[450:] #membuat data testing dari hasil
  pengurangan 500-450

```

| | | | |
|---------------|-----------|-----------|---|
| data_testing | DataFrame | (50, 12) | Column names: first_name, last_name, company_name, address, city, coun ... |
| data_training | DataFrame | (450, 12) | Column names: first_name, last_name, company_name, address, city, coun ... |

Gambar 4.73 pecahan dataframe

3. No. 3

```

1 # In [1]:
2 import pandas as pd #Mengimport pandas
3 d=pd.read_csv("D:/Git Kecerdasan Buatan/KB3C/src/1174084/4/
  Youtube02-KatyPerry.csv") #Membuat variable d untuk
  membaca file csv dari dataset
4 # In [2]:
5 spam=d.query('CLASS == 1') #mengelompokkan komentar spam
6 nospam=d.query('CLASS == 0') #mengelompokkan komentar bukan
  spam
7 # In [3]: memanggil lib vektorisasi
8 #melakukan fungsi bag of word dengan cara menghitung semua
  kata
9 #yang terdapat dalam file
10 from sklearn.feature_extraction.text import CountVectorizer
11 vectorizer = CountVectorizer()
12 # In [3]:
13 dvec = vectorizer.fit_transform(d['CONTENT']) #melakukan bag
  of word pada dataframe pada colom CONTENT
14 # In [4]:
15 dvec #melihat isi vektorisasi
16 # In [5]:
17 print(d['CONTENT'][342]) #melihat isi data pada baris ke 342
18 # In [6]:

```

```
19 daptarkata=vectorizer.get_feature_names() #feature_names  
    merupakan digunakan untuk mengambil nama kolomnya ada apa  
    saja  
20 # In [7]:  
21 dshuf = d.sample(frac=1) #melakukan randomisasi pada datanya  
    supaya sempurna saat melakukan klasifikasi  
22 # In [8]:  
23 dk_train=dshuf[:300] #Data akan dibagi dari 300 row akhir  
    menjadi data training dan sisanya adalah data testing  
24 dk_test=dshuf[300:] #Data akan dibagi dari 300 row pertama  
    menjadi data training dan sisanya adalah data testing  
25 # In [9]:  
26 dk_train_att=vectorizer.fit_transform(dk_train['CONTENT']) #  
    melakukan training pada data training dan di vektorisasi  
27 print(dk_train_att)  
28 # In [10]:  
29 dk_test_att=vectorizer.transform(dk_test['CONTENT']) #  
    melakukan testing pada data testing dan di vektorisasi  
30 print(dk_test_att)  
31 # In [11]:  
32 dk_train_label=dk_train['CLASS'] #mengambil label spam dan  
    bukan spam  
33 print(dk_train_label)  
34 dk_test_label=dk_test['CLASS'] #mengambil label spam dan  
    bukan spam  
35 print(dk_test_label)
```

| | |
|-----------|---|
| (0, 336) | 1 |
| (0, 242) | 1 |
| (0, 1561) | 1 |
| (0, 733) | 1 |
| (0, 1029) | 1 |
| (0, 1178) | 1 |
| (0, 1078) | 1 |
| (0, 811) | 1 |
| (1, 495) | 1 |
| (1, 1318) | 1 |
| (1, 816) | 1 |
| (1, 74) | 1 |
| (1, 479) | 1 |
| (1, 111) | 1 |
| (1, 113) | 1 |
| (1, 1256) | 1 |
| (1, 425) | 1 |
| (1, 1285) | 1 |
| (1, 180) | 6 |
| (1, 741) | 1 |
| (1, 399) | 2 |
| (1, 241) | 2 |

Gambar 4.74 vektorisasi dan klasifikasi

4. No. 4

```

1 # In[12]:
2 from sklearn import svm #Mengimport svm
3 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
4         fungsi SVC
5 clfsvm.fit(dk_train_att, dk_train_label) #Mengatur data
6         training
7 clfsvm.predict(dk_test_att)
8 clfsvm.score(dk_test_att, dk_test_label) #Mengatur data
9         testing

```

```

In [62]: clfsvm.predict(dk_test_att)
Out[62]:
array([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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0], dtype=int64)

In [63]: clfsvm.score(dk_test_att, dk_test_label) #Mengatur data testing
Out[63]: 0.5

```

Gambar 4.75 klasifikasi SVM

5. No. 5

```

1 # In[13]:
2 from sklearn import tree #Mengimport tree
3 clftree = tree.DecisionTreeClassifier() #clftree sebagai
4         variabel untuk decision tree
5 clftree.fit(dk_train_att, dk_train_label) #Mengatur data
6         training
7 # In[14]:
8 clftree.predict(dk_test_att)
9 # In[15]:
10 clftree.score(dk_test_att, dk_test_label) #Mengatur data
11         testing

```

```

In [60]: clftree.predict(dk_test_att)
Out[60]:
array([0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 1, 1], dtype=int64)

In [61]: clftree.score(dk_test_att, dk_test_label) #Mengatur data testing
Out[61]: 0.92

```

Gambar 4.76 klasifikasi Decission Tree

6. No. 6

```

1 # In[16]:
2 from sklearn.metrics import confusion_matrix #Mengimport
3         Confusion Matrix

```

```

3 pred_labels = clf.predict(dk_test_att) #Membuat variable
    pred_labels dari data testing
4 cm = confusion_matrix(dk_test_label, pred_labels) #cm sebagai
    variabel data label
5 cm

```

Out[65]:

```
array([[24,  0],
       [ 3, 23]], dtype=int64)
```

Gambar 4.77 confusion matrix

7. No. 7

```

1 # In[17]:
2 from sklearn.model_selection import cross_val_score #
    Mengimport cross_val_score
3 scores = cross_val_score(clf,dk_train_att,dk_train_label,cv
    =5) #Membuat variable scores sebagai variabel prediksi
        dari data training
4 scorerata2=scores.mean()
5 scorersd=scores.std()
6 # In[18]
7 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
    std() * 2)) #menampilkan data scores dengan ketentuan
        akurasi

```

```

In [66]: from sklearn.model_selection import cross_val_score #Mengimport
cross_val_score
...: scores = cross_val_score(clf,dk_train_att,dk_train_label,cv=5) #Membuat
variable scores sebagai variabel prediksi dari data training
...: scorerata2=scores.mean()
...: scorersd=scores.std()

In [67]: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
#menampilkan data scores dengan ketentuan akurasi
Accuracy: 0.95 (+/- 0.06)

```

Gambar 4.78 cross validaiton

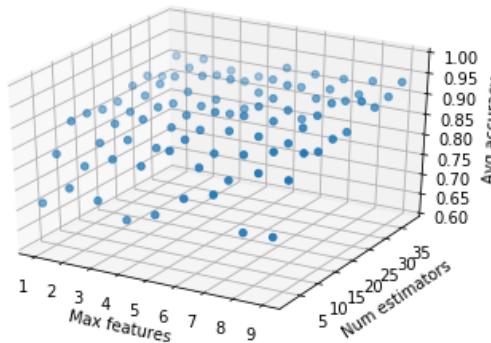
8. No. 8

```

1
2 # In[19]:
3 import numpy as np
4 max_features_opts = range(1, 10, 1) #Variable
    max_features_opts sebagai variabel untuk membuat range 1,
        10, 1

```

```
5 n_estimators_opts = range(2, 40, 4) #
  Variable n_estimators_opts sebagai variabel untuk membuat
  range 2, 40, 4
6 rf_params = np.empty((len(max_features_opts)*len(
  n_estimators_opts),4) , float) #Variable rf_params sebagai
  variabel untuk menjumlahkan yang sudah di tentukan
  sebelumnya
7 i = 0
8 for max_features in max_features_opts: #Perulangan
9   for n_estimators in n_estimators_opts: #Perulangan
10     clf = RandomForestClassifier(max_features=
11       max_features, n_estimators=n_estimators) #Menampilkan
12       variabel csf
13       scores = cross_val_score(clf , dk_train_att ,
14       dk_train_label , cv=5) #Variable scores sebagai variabel
15       training
16       rf_params[i,0] = max_features #index 0
17       rf_params[i,1] = n_estimators #index 1
18       rf_params[i,2] = scores.mean() #index 2
19       rf_params[i,3] = scores.std() * 2 #index 3
20       i += 1 #Dengan ketentuan i += 1
21       print("Max features: %d, num estimators: %d, accuracy
22 : %0.2f (+/- %0.2f)" %
23 (% (max_features , n_estimators , scores.mean() , scores.std() *
24 2))) #Print hasil pengulangan yang sudah ditentukan
25
26 # In [20]:
27 import matplotlib.pyplot as plt #Mengimport library
28   matplotlib sebagai plt
29 from mpl_toolkits.mplot3d import Axes3D #Mengimport axes3D
30   untuk menampilkan plot 3 dimensi
31 from matplotlib import cm #Memanggil data cm yang sudah
32 tersedia
33 fig = plt.figure() #Menghasilkan plot sebagai figure
34 fig.clf() #Figure di ambil dari clf
35 ax = fig.gca(projection='3d') #ax sebagai projection 3d
36 x = rf_params[:,0] #x sebagai index 0
37 y = rf_params[:,1] #y sebagai index 1
38 z = rf_params[:,2] #z sebagai index 2
39 ax.scatter(x, y, z) #Membuat plot scatter x y z
40 ax.set_zlim(0.6, 1) #Set zlim dengan ketentuan yang ada
41 ax.set_xlabel('Max features') #Memberikan nama label x
42 ax.set_ylabel('Num estimators') #Memberikan nama label y
43 ax.set_zlabel('Avg accuracy') #Memberikan nama label z
```



Gambar 4.79 pengamatan komponen informasi

4.6.3 Penanganan Error

1. Error

```
FileNotFoundException: [Errno 2] File b'D:/Git Kecerdasan Buatan/KB3C/src/11174084/4/Youtube02-KatyPerry.csv' does not exist: b'D:/Git Kecerdasan Buatan/KB3C/src/11174084/4/Youtube02-KatyPerry.csv'
```

Gambar 4.80 File Not Found Error

2. Tuliskan Kode Error dan Jenis Error

- FileNotFoundException

3. Cara Penanganan Error

- FileNotFoundException

Error terdapat pada kesalahan saat baca file csv, yang tidak terbaca. Dikarenakan letak file yang dibaca tidak pada direktori yang sama. Seharusnya letakkan file di direktori yang sama.

4.6.4 Bukti Tidak Plagiat



Gambar 4.81 plagiarism

4.6.5 Link Video Youtube

https://youtu.be/v63ayQTw_MI

4.7 1174077 - Alvan Alvanzah

4.7.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.82 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri.

Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.83 Klasifikasi Bunga.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri. Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya

tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.84 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.

Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.85 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.86 TF-IDF.

4.7.2 Praktek Program

1. Soal 1

```

1 #%% Soal 1
2 import pandas as pd #digunakan untuk mengimport library
# pandas dengan alias pd
3 pd = pd.read_csv("C:/Users/ASUS/Videos/zuplur/src/1174077/4/
#membaca file csv

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :



Gambar 4.87 Hasil Soal 1.

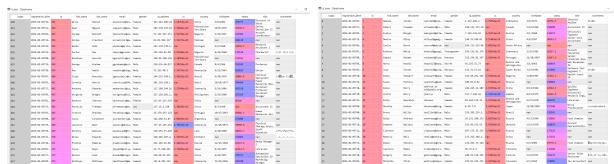
2. Soal 2

```

1 #%% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisanya dari data
# yang tersedia

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :



Gambar 4.88 Hasil Soal 2.

3. Soal 3

```

1 %% Soal 3
2 import pandas as ps #untuk import library pandas berguna
    untuk mengelola dataframe
3 ps = ps.read_csv("C:/Users/ASUS/Videos/zuplur/src/1174077/4/
    Youtube03-LMFAO.csv") #membaca file dengan format csv
4
5 spam=ps.query( 'CLASS == 1') #membagi tabel spam
6 nospam=ps.query( 'CLASS == 0')#membagi tabel no spam
7
8 from sklearn.feature_extraction.text import CountVectorizer #
    untuk import countvectorizer berfungsi untuk memecah data
    tersebut menjadi sebuah kata yang lebih sederhana
9 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
    tersebut, pada code ini tidak ada hasilnya dikarenakan
    spyder tidak mendukung hasil dari instasiasi.
10
11 dvec = vectorizer.fit_transform(ps[ 'CONTENT']) #untuk
    melakukan pemecahan data pada dataframe yang terdapat pada
    kolom konten
12 dvec #Untuk menampilkan hasil dari code sebelumnya
13
14 Daptarkata= vectorizer.get_feature_names()
15
16 dshuf = ps.sample(frac=1)
17
18 d_train=dshuf[:300]
19 d_test=dshuf[300:]
20
21 d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT'])
22 d_train_att
23
24 d_train_label=d_train[ 'CLASS']
25 d_test_label=d_test[ 'CLASS']
```

Dengan menggunakan $1174077 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

| Index | COMMENT_ID | AUTHOR | DATE | CONTENT | CLASS |
|-------|----------------|---------------|------------------|----------------------------------|-------|
| 0 | 1120w0hepd.. | Corey Wilson | 2015-05-28T23... | (a) I'm not trying to offend but | 0 |
| 1 | 1124ycrczJd.. | Taisi Goming | 2015-05-28T23... | hey guys, I have | 1 |
| 2 | 1127tgcjyfJh.. | Lek Music | 2015-05-28T23... | Part of the Rock...lol... | 0 |
| 3 | 1127trDm9pa.. | Cheryl Fox | 2015-05-28T23... | Rock...lol... | 0 |
| 4 | 1128px4kx4ad.. | PATRICK TU | 2015-05-28T23... | Party Rock | 0 |
| 5 | 1129myewew.. | Brian Brad | 2015-05-28T23... | Surprise | 0 |
| 6 | 1129ed11oy.. | Brian Brad | 2015-05-28T23... | | 0 |
| 7 | 1131100mog.. | Alex Dafeo | 2015-05-28T23... | This song is | 0 |
| 8 | 1131gmc6tl.. | Gloward | 2015-05-28T23... | Just really... | 0 |
| 9 | 1131mcnzm.. | Jessnet | 2015-05-28T23... | awesomeness :) | 0 |
| 10 | 1131mcnzm.. | Silvia Basur | 2015-05-28T23... | Incredible song | 0 |
| 11 | 1132ctvYrd.. | Uniteaside | 2015-05-28T23... | song so much | 0 |
| 12 | 1134g45t4u.. | Notified | 2015-05-28T23... | 2015 Lireff | 0 |
| 13 | 1135trxtetP.. | Alex Martin | 2015-05-28T23... | people dress, | 0 |
| 14 | 1136u2ztrv.. | Alex Jensen | 2015-05-28T23... | last year of, | 0 |
| 15 | 1136w1x3m.. | Aiden Hill | 2015-05-27T22... | Best song | 0 |
| 16 | 1137rtsgipz.. | [Joe penti] | 2015-05-27T22... | super nice, | 0 |
| 17 | 1137rgjfxk.. | Silvia Kewo | 2015-05-27T22... | ooooooooooooo | 0 |
| 18 | 1137rgjfxk.. | SoulInTheHaus | 2015-05-27T21... | PARTY ROCK | 0 |
| 19 | 113dy5gnqec.. | Phuong Lin | 2015-05-27T21... | Thumbs up if | 0 |
| 20 | 113ef7sp9om.. | Goncalo | 2015-05-27T21... | this is | 0 |
| | 113ew4qro6.. | katrin qpk | 2015-05-27T21... | LMAO!!!!!! | 0 |

Gambar 4.89 Hasil Soal 3.

4. Soal 4

```
1 #%% Soal 4
2
3 from sklearn import svm
4 clfsvm = svm.SVR(gamma = 'auto')
5 clfsvm.fit(d_train_att , d_train_label)
```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```
In [7]: from sklearn import svm
... clfsvm = svm.SVR(gamma = 'auto')
... clfsvm.fit(d_train_att , d_train_label)
Out[7]: SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1,
         gamma='auto', kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
         verbose=False)
```

Gambar 4.90 Hasil Soal 4.

5. Soal 5

```
1 #%%soal 5
2
3 from sklearn import tree
4 clftree = tree.DecisionTreeClassifier()
5 clftree.fit(d_train_att , d_train_label)
```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut :

```
In [8]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
Out[8]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated', random_state=None, splitter='best')
```

Gambar 4.91 Hasil Soal 5.

6. Soal 6

```
1 #%%soal 6/
2
3 from sklearn.metrics import confusion_matrix
4 pred_labels=clftree.predict(d_test)
5 cm=confusion_matrix(d_test_label ,pred_labels)
6
7 #%%
8
9 import matplotlib.pyplot as plt
10
11 def plot_confusion_matrix(cm, classes,
12                           normalize=False,
13                           title='Confusion matrix',
14                           cmap=plt.cm.Blues):
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
18         print("Normalized confusion matrix")
19     else:
20         print('Confusion matrix, without normalization')
21
22     print(cm)
```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...:
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:     cm
```

Gambar 4.92 Hasil Soal 6.

7. Soal 7

```
1 #%%soal 7
2
```

```

3 from sklearn.model_selection import cross_val_score
4
5 scores=cross_val_score(clftree,d_train_att,d_train_label, cv
6 =5)
7
8 skor_rata2=scores.mean()
9 skoresd=scores.std()

```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```

In [12]: from sklearn.model_selection import cross_val_score
...:
scores=cross_val_score(clftree,d_train_att,d_train_label, cv=5)
...:
skor_rata2=scores.mean()
...:
skoresd=scores.std()

```

Gambar 4.93 Hasil Soal 7.

8. Soal 8

```

1 %%soal 8 /
2
3 max_features_opts = range(5, 50, 5) #max_features_opts
4 sebagai variabel untuk membuat range 5,50,5
5 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
6 sebagai variabel untuk membuat range 10,200,20
7 rf_params = ps.empty((len(max_features_opts)*len(
8 n_estimators_opts),4), float) #rf_params sebagai variabel
9 untuk menjumlahkan yang sudah di tentukan sebelumnya
10 i = 0
11 for max_features in max_features_opts: #pengulangan
12     for n_estimators in n_estimators_opts: #pengulangan
13         clftree = RandomForestClassifier(max_features=
14             max_features, n_estimators=n_estimators) #menampilkan
15         variabel csf
16         scores = cross_val_score(clf, df_train_att,
17             df_train_label, cv=5) #scores sebagai variabel training
18         rf_params[i,0] = max_features #index 0
19         rf_params[i,1] = n_estimators #index 1
20         rf_params[i,2] = scores.mean() #index 2
21         rf_params[i,3] = scores.std() * 2 #index 3
22         i += 1 #dengan ketentuan i += 1
23         print("Max features: %d, num estimators: %d, accuracy
24 : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
25 mean(), scores.std() * 2))
26         #print hasil pengulangan yang sudah ditentukan

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```
Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 100, accuracy: 0.33 (+/- 0.02)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 30, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 30, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.34 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 100, accuracy: 0.34 (+/- 0.02)
Max features: 40, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 40, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 40, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 40, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 100, accuracy: 0.34 (+/- 0.02)
Max features: 45, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
```

Gambar 4.94 Hasil Soal 8.

4.7.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundError: [Errno 2] File 'E:\Users\ALVAN\Downloads\taylor\src\1174077\4\alvaca.csv' does
not exist: 'E:\Users\ALVAN\Downloads\taylor\src\1174077\4\alvaca.csv'
```

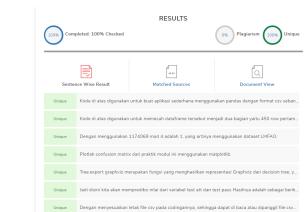
Gambar 4.95 SyntaxError

2. Cara Penanganan Error

- SyntaxError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.7.4 Bukti Tidak Plagiat

**Gambar 4.96** Bukti Tidak Melakukan Plagiat Chapter 4

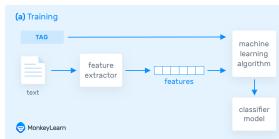
4.7.5 Link Youtube

4.8 1174079 -Chandra Kirana Poetra

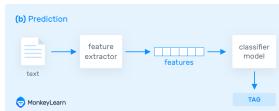
4.8.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan suatu proses dalam memberikan tag atau kategori pada suatu teks berdasarkan konten. Klasifikasi teks merupakan salah satu bagian dari natural language processing yang mencakup analisis, labeling, deteksi spam dan intent detection. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.97 Klasifikasi teks.



Gambar 4.98 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri. Karena meskipun bunga yang kita coba untuk klasifikasikan memiliki spesies yang sama, kebanyakan bunga sendiri memiliki bentuk ukuran yang berbeda beda sehingga akan membuat machine learning sulit diterapkan karena ukuran tidak bisa dijadikan sebagai salah satu parameter dalam machine learning.



Gambar 4.99 Ukuran Bunga yang berbeda-beda.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.
Youtube menggunakan istilah yang disebut sebagai keyword yang nantinya kita isi didalam form pencarian yang ada kemudian akan muncul video yang terkait



Gambar 4.100 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.
vektorisasi merupakan suatu proses untuk mengkonversi suatu algoritma yang beroperasi pada satu value dalam satu waktu menjadi kebeberapa value dalam satu waktu
5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Merupakan model untuk menyederhanakan data dalam bidang natural language processing dan juga pengambilan informasi. di model ini text direpresentasikan sebagai suatu kantong kata, mengabaikan grammer dan susunan kata tapi menyimpan value pengulangan apabila ada kata yang diulang. model ini digunakan pada bidang komputer



Gambar 4.101 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan singkatan dari term frequency inverse document frequency yang merupakan statistik angka yang memiliki tujuan untuk memperlihatkan seberapa pentingnya suatu kata dalam suatu dokumen.



Gambar 4.102 TF-IDF.

4.8.2 Praktek Program

4.8.2.1 Nomor 1

```
1 %% Soal 1
2 import pandas as pd #digunakan untuk mengimport library pandas
3 dengan alias pd
3 pd = pd.read_csv("F:/ Poltekpos/D4 TI 3C/Semester 6/Kecerdasan
4 Buatan/Github/Upload 30 Maret 2020 github tugas 4/src
5 /1174079/4/csv_chandra.csv") #membaca file csv
```

```
In [32]: import pandas as pd #digunakan untuk mengimport
library pandas dengan alias pd
... pd = pd.read_csv("F:/Poltekpos/D4 TI 3C/Semester 6/
Kecerdasan/Buatan/Github/Upload 30 Maret 2020 github tugas 4/
src/1174079/4/csv_chandra.csv") #membaca file csv
```

Gambar 4.103 Nomor 1

4.8.2.2 Nomor 2

```
1 %% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data yang
tersedia
```

```
In [33]: d_train=pd[:450] #membagi data training menjadi 450
... d_test=pd[450:] #membagi data menjadi 50 atau sisa
dari data yang tersedia
```

Gambar 4.104 Nomor 2

4.8.2.3 Nomor 3

```
1 %% Soal 3
2 import pandas as pd #digunakan untuk mengimport library pandas
3 dengan alias pd
```

```

3 d = pd.read_csv("F:/ Poltekpos/D4 TI 3C/Semester 6/Kecerdasan
    Buatan/Github/Upload 30 Maret 2020 github tugas 4/src
    /1174079/4/Youtube03-LMFAO.csv") #Membaca file csv
4
5 from sklearn.feature_extraction.text import CountVectorizer #
    import fungsi_countvectorize dari sklearn
6 vectorizer = CountVectorizer() #membuat instansi CountVectorizer
7
8 dvec = vectorizer.fit_transform(d['CONTENT']) #Memasukkan data ke
    dvec
9 dvec #Melihat data yang dimasukkan ke dvec
10
11 daptarkata = vectorizer.get_feature_names() #Mendapatkan data dan
    memasukkannya ke daptarkata
12
13 dshuf = d.sample(frac=1) #Memasukkan sample kedalam variable
    dshuf
14
15 d_train = dshuf[:300] #Membuat data training
16 d_test = dshuf[300:] #Membuat data test
17
18 d_train_att = vectorizer.fit_transform(d_train['CONTENT']) #
    Memasukkan data training dari vectorizer
19 d_train_att #Melihat data training
20
21 d_test_att = vectorizer.transform(d_test['CONTENT']) #Memasukkan
    data test dari vectorizer
22 d_test_att #Melihat data training
23
24 d_train_label = d_train['CLASS'] #Memberi label
25 d_test_label = d_test['CLASS'] #Memberi Label

```

```

rc(1174079/4/Youtube03-LMFAO.csv") #Membaca file csv
...
...     from sklearn.feature_extraction.text import
        CountVectorizer #import fungsi countvectorize dari sklearn
...     vectorizer = CountVectorizer() #membuat instansi
        CountVectorizer
...
...     dvec = vectorizer.fit_transform(d['CONTENT'])
        Memasukkan data ke dvec
...
...     dvec #Melihat data yang dimasukkan ke dvec
...
...     daptarkata = vectorizer.get_feature_names()
        Mendapatkan data dan memasukkannya ke daptarkata
...
...     dshuf = d.sample(frac=1) #Memasukkan sample kedalam
        variable dshuf
...
...     d_train = dshuf[:300] #Membuat data training
...     d_test = dshuf[300:] #Membuat data test
...
...     d_train_att =
        vectorizer.fit_transform(d_train['CONTENT']) #Memasukkan
        training dari vectorizer
...     d_train_att #Melihat data training
...
...     d_test_att = vectorizer.transform(d_test['CONTENT'])
        Memasukkan data test dari vectorizer
...     d_test_att #Melihat data training
...
...     d_train_label = d_train['CLASS'] #Memberi label
        d_train_label = d_train['CLASS'] #Memberi label

```

Gambar 4.105 Nomor 3

4.8.2.4 Nomor 4

```

1 # SVM
2 from sklearn import svm #Mengimport svm dari sklearn
3 clfsvm = svm.SVC() #Membuat svc kedalam variable svm

```

```

4 clfsvm.fit(d_train_att, d_train_label) #Memprediksi data dari
   data training
5 clfsvm.score(d_test_att, d_test_label) #Memunculkan clf sebagai
   testing yang sudah di training tadi

```

```

In [38]: clfsvm.fit(d_train_att, d_train_label)
          C:\Users\acer\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
              from 'auto' to 'scale' in version 0.22 to account better for
              unbalanced classes; specify gamma explicitly to 'auto' or 'scale'
              to avoid this warning.
              "avoid this warning.", FutureWarning)
Out[38]: 0.797104492753623

```

Gambar 4.106 Nomor 4

4.8.2.5 Nomor 5

```

1 # Decission Tree
2 from sklearn import tree #Mengimport tree dari sklearn
3 clftree = tree.DecisionTreeClassifier() #Membuat decision tree
4 clftree.fit(d_train_att, d_train_label) #Memprediksi data dari
   data training
5 clftree.score(d_test_att, d_test_label) #Memunculkan clf sebagai
   testing yang sudah di training tadi

```

```

In [49]: from sklearn import tree #Mengimport tree dari
          sklearn
          ...
          clftree = tree.DecisionTreeClassifier() #Membuat
          decision tree
          ...
          clftree.fit(d_train_att, d_train_label) #Memprediksi
          data dari data training
          ...
          clftree.score(d_test_att, d_test_label) #Memunculkan
          clf sebagai testing yang sudah di training tadi
Out[49]: 0.9492753623188406

```

Gambar 4.107 Nomor 5

4.8.2.6 Nomor 6

```

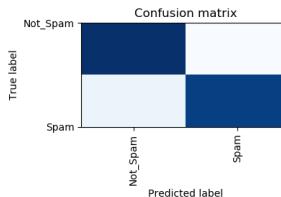
1 # Confusion Matrix – Decission Tree
2 from sklearn.metrics import confusion_matrix
3 pred_labeltree = clftree.predict(d_test_att)
4 cmtree = confusion_matrix(d_test_label, pred_labeltree)
5 cmtree
6
7 import matplotlib.pyplot as plt
8 import itertools
9 def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues):
10
11     """
12     This function prints and plots the confusion matrix.
13     Normalization can be applied by setting 'normalize=True'.
14     """
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
18         print("Normalized confusion matrix")
19
20     else:

```

```

21     print('Confusion matrix, without normalization')
22
23     print(cm)
24
25     plt.imshow(cm, interpolation='nearest', cmap=cmap)
26     plt.title(title)
27     #plt.colorbar()
28     tick_marks = np.arange(len(classes))
29     plt.xticks(tick_marks, classes, rotation=90)
30     plt.yticks(tick_marks, classes)
31
32     fmt = '.2f' if normalize else 'd'
33     thresh = cm.max() / 2.
34     #for i, j in itertools.product(range(cm.shape[0]), range(cm.
35     #shape[1])):
36     #    plt.text(j, i, format(cm[i, j], fmt),
37     #              horizontalalignment="center",
38     #              color="white" if cm[i, j] > thresh else "black"
39     #)
40
41     plt.tight_layout()
42     plt.ylabel('True label')
43     plt.xlabel('Predicted label')
44
45 types = pd.read_csv("F:/Poltekpos/D4 TI 3C/Semester 6/Kecerdasan
46 Buatan/Github/Upload 30 Maret 2020 github tugas 4/src
47 /1174079/4/classes.txt", sep='|', header=None, usecols=[1],
48 names=['type'])
49 types = types['type']
50 types
51
52 import numpy as np
53 np.set_printoptions(precision=2)
54 plt.figure(figsize=(4,4), dpi=100)
55 plot_confusion_matrix(cmtree, classes=types, normalize=True)
56 plt.show()

```



Gambar 4.108 Nomor 6

4.8.2.7 Nomor 7

```

1 # Cross Validation
2 from sklearn.model_selection import cross_val_score
3

```

```

4 scorestree = cross_val_score(clftree, d_train_att, d_train_label,
5     cv=5)
6 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
7     scorestree.std() * 2))
8
9 scoressvm = cross_val_score(clfsvm, d_train_att, d_train_label,
10    cv=5)
11 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
12     scoressvm.std() * 2))
13
14 scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
15 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
16     () * 2))

```

```
C:\Users\acer\OneDrive\Udemy\stephanie\sklearn\svmbase.py:103: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for
unscaled features. Set gamma explicitly to 'auto' or 'scale'
to avoid this warning.
  "Avoid this warning.", FutureWarning)
Accuracy: 0.95 (+/- 0.00)
```

Gambar 4.109 Nomor 7

4.8.2.8 Nomor 8

```

1 # Komponen Informasi
2 max_features_opts = range(5, 50, 5)
3 n_estimators_opts = range(10, 200, 20)
4 rf_params = np.empty((len(max_features_opts)*len(
5     n_estimators_opts),4), float)
6 i = 0
7 for max_features in max_features_opts:
8     for n_estimators in n_estimators_opts:
9         clf = RandomForestClassifier(max_features=max_features,
10             n_estimators=n_estimators)
11         scores = cross_val_score(clf, d_train_att, d_train_label,
12             cv=5)
13         rf_params[i,0] = max_features
14         rf_params[i,1] = n_estimators
15         rf_params[i,2] = scores.mean()
16         rf_params[i,3] = scores.std() * 2
17         i += 1
18         print("Max features: %d, num estimators: %d, accuracy:
19             %0.2f (+/- %0.2f)" % (max_features, n_estimators, scores.mean(
20                 ), scores.std() * 2))      #print hasil pengulangan yang
21         sudah ditentukan

```

```

usage: [options] [files...]
Max features: 5, num estimators: 10, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 30, accuracy: 0.92 (+/- 0.07)
Max features: 5, num estimators: 50, accuracy: 0.94 (+/- 0.04)
Max features: 5, num estimators: 70, accuracy: 0.94 (+/- 0.06)
Max features: 5, num estimators: 90, accuracy: 0.95 (+/- 0.05)
Max features: 5, num estimators: 110, accuracy: 0.95 (+/- 0.04)
Max features: 5, num estimators: 130, accuracy: 0.95 (+/- 0.04)
Max features: 5, num estimators: 150, accuracy: 0.94 (+/- 0.08)
Max features: 5, num estimators: 170, accuracy: 0.93 (+/- 0.05)
Max features: 5, num estimators: 190, accuracy: 0.94 (+/- 0.07)
Max features: 10, num estimators: 10, accuracy: 0.92 (+/- 0.03)
Max features: 10, num estimators: 30, accuracy: 0.95 (+/- 0.04)
Max features: 10, num estimators: 50, accuracy: 0.93 (+/- 0.07)
Max features: 10, num estimators: 70, accuracy: 0.93 (+/- 0.07)
Max features: 10, num estimators: 90, accuracy: 0.94 (+/- 0.06)
Max features: 10, num estimators: 110, accuracy: 0.93 (+/- 0.06)

```

Gambar 4.110 Nomor 8

4.8.3 Penanganan Error

1. ScreenShoot Error

```

FileNotFoundException: [Error 3] File b'1174087/4/github/04 TI 3C Semester 6/kecerdasan Buata/Github/upload 30 Maret 2020 github tugas 4/src/1174079/4/csv_chandra.csv' does not exist: b'F:\Poltekpos\04 TI 3C\Semester 6\kecerdasan Buata/Github\Upload 30 Maret 2020 github tugas 4/src/1174079/4/csv_chandra.csv'

```

Gambar 4.111 SyntaxError

2. Cara Penangan Error

- **SyntaxError**

Memperbaiki typo dan mengecek nama file serta direktorinya

4.8.4 Bukti Tidak Plagiat

**Gambar 4.112** Bukti Tidak Melakukan Plagiat Chapter 4

4.8.5 Link Youtube

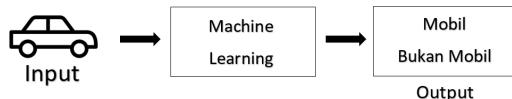
<https://youtu.be/8vkMpgUVTQs>

4.9 1174087 - Ilham Muhammad Ariq

4.9.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

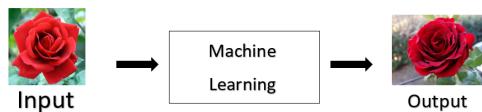
Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut



Gambar 4.113 Klasifikasi teks

2. Jelaskan mengapa klasifikasi bunga tidak bisa menggunakan machine learning, sertakan ilustrasi sendiri.

Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun bentuk dan ukurannya tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut



Gambar 4.114 Klasifikasi Bunga

3. Jelaskan bagaimana teknik pembelajaran mesin pada teks pada kata-kata yang digunakan di youtube,jelaskan arti per atribut data csv dan sertakan ilustrasi buatan sendiri.

Teknik machine learning yang dipakai pada teks yang digunakan di youtube bisa menggunakan bag of words dan random forest. Bag of words adalah proses mengubah teks menjadi vektor dengan panjang tetap dengan cara menghitung berapa kali setiap kata itu muncul. Random forest (RF) adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon (tree) dengan melakukan training pada sampel data yang dimiliki.

Atribut yang ada pada file csv Youtube01-Psy diantaranya, COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS.

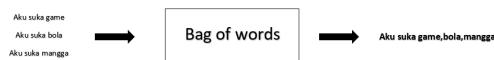
- COMMENT_ID : merupakan key unik yang membedakan komen lainnya.
- AUTHOR : merupakan penulis dari komen tersebut.
- DATE : merupakan waktu dari komen tersebut dipublikasikan.
- CONTENT : merupakan isi komentarnya.
- CLASS : merupakan klasifikasi dari komennya (spam/notspam).

4. Jelaskan apa yang dimaksud vektorisasi data.

Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Bag of words adalah representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut



Gambar 4.115 Bag of words

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut

| sentence 1 | earth is the third planet from the sun | | TF*IDF (sentence 1) | TF*IDF (Sentence 2) |
|------------|--|------------------|---------------------|---------------------|
| sentence 2 | Jupiter is the largest planet | | | |
| Word | TF (Sentence 1) | TF (Sentence 2) | TF*IDF (sentence 1) | TF*IDF (Sentence 2) |
| earth | 1/8 | 0 log(2/1)=0 | 0.0375 | 0 |
| is | 1/8 | 1/5 log(2/2)=0 | 0 | 0 |
| the | 2/8 | 1/5 log(2/2)=0 | 0 | 0 |
| third | 1/8 | 0 log(2/1)=0.3 | 0.0375 | 0 |
| planet | 1/8 | 1/5 log(2/2)=0 | 0 | 0 |
| from | 0 | 0 log(2/1)=0.3 | 0 | 0 |
| sun | 1/8 | 0 log(2/1)=0.3 | 0.0375 | 0 |
| largest | 0 | 1/5 log(2/1)=0.3 | 0 | 0.06 |
| Jupiter | 0 | 1/5 log(2/1)=0.3 | 0 | 0.06 |

Gambar 4.116 TF-IDF

4.9.2 Praktek Program

1. Soal 1

```
1 import pandas as pd #digunakan untuk mengimport library
    pandas dengan alias pd
2 pd = pd.read_csv("E:/Kecerdasan Buatan/KB3C/src/1174087/4/
    csv_ariq.csv") #membaca file csv
```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut

```
In [146]: import pandas as pd #digunakan untuk mengimport library pandas dengan
alias pd
...: pd = pd.read_csv("E:/Kecerdasan Buatan/KB3C/src/1174087/4/csv_ariq.csv")
membaca file csv
```

Gambar 4.117 Soal 1

2. Soal 2

```
1 d_train=pd[:450] #membagi data training menjadi 450
2 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data
    yang tersedia
```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut

```
In [147]: d_train=pd[:450] membagi data training menjadi 450
...: d_test=pd[450:] membagi data menjadi 50 atau sisa dari data yang
tersedia
```

Gambar 4.118 Soal 2

3. Soal 3

```
1 import pandas as pd #untuk import library pandas berguna
    untuk mengelola dataframe
2 ariq = pd.read_csv("E:/Kecerdasan Buatan/KB3C/src/1174087/4/
    Youtube05-Shakira.csv") #membaca file dengan format csv
```

```

3
4 from sklearn.feature_extraction.text import CountVectorizer #
    untuk import countvectorizer berfungsi untuk memecah data
    tersebut menjadi sebuah kata yang lebih sederhana
5 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
    tersebut, pada code ini tidak ada hasilnya dikarenakan
    spyder tidak mendukung hasil dari instasiasi.
6
7 dvec = vectorizer.fit_transform(ariq[ 'CONTENT' ]) #untuk
    melakukan pemecahan data pada dataframe yang terdapat pada
    kolom konten
8 dvec #Untuk menampilkan hasil dari code sebelumnya
9
10 Daptarkata= vectorizer.get_feature_names()
11
12 dshuf = ariq.sample(frac=1)
13
14 d_train=dshuf[:300]
15 d_test=dshuf[300:]
16
17 d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT' ])
d_train_att
18
19 d_test_att = vectorizer.transform(d_test[ 'CONTENT' ])
d_test_att
20
21 d_train_label=d_train[ 'CLASS' ]
22 d_test_label=d_test[ 'CLASS' ]

```

Dengan menggunakan $1174087 \bmod 4$ adalah 3, yang artinya menggunakan dataset shakira. Hasilnya adalah sebagai berikut

```

In [148]: Import pandas as pd #untuk import library pandas berguna untuk mengelola
          data
...: ariq = pd.read_csv("E:/Kecerdasan Buatan/X83C/src/1174087/4/youtube05-
Shakira.csv") #memuat file dengan format csv
...:
...: from sklearn.feature_extraction.text import CountVectorizer #untuk import
countvectorizer fungsi untuk memecah data tersebut menjadi sebuah kata yang
lebih sederhana
...: vectorizer = CountVectorizer () #ntuk menjalankan fungsi tersebut, pada
code ini tidak ada hasilnya dikarenakan spyder tidak mendukung hasil dari
instasiasi.
...:
...: dvec = vectorizer.fit_transform(ariq[ 'CONTENT' ]) #untuk melakukan
pemecahan data pada dataframe yang terdapat pada kolom konten
...: dvec #Untuk menampilkan hasil dari code sebelumnya
...:
...: Daptarkata= vectorizer.get_feature_names()
...:
...: dshuf = ariq.sample(frac=1)
...:
...: d_train=dshuf[:300]
...: d_test=dshuf[300:]
...:
...: d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT' ])
d_train_att
...:
...: d_test_att = vectorizer.transform(d_test[ 'CONTENT' ])
d_test_att
...:
...: d_train_label=d_train[ 'CLASS' ]
...: d_test_label=d_test[ 'CLASS' ]

```

Gambar 4.119 Soal 3

4. Soal 4

```

1 from sklearn import svm
2 clfsvm = svm.SVC()
3 clfsvm.fit(d_train_att , d_train_label)

```

Klasifikasi dari data vektorisasi menggunakan klasifikasi SVM. Hasilnya adalah sebagai berikut

```
In [150]: from sklearn import svm
.....
Out[150]:
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
kernel='rbf', max_iter=-1, probability=False, random_state=None,
shrinking=True, tol=0.001, verbose=False)
```

Gambar 4.120 Soal 4

5. Soal 5

```
1 from sklearn import tree
2 clftree = tree.DecisionTreeClassifier()
3 clftree.fit(d_train_att, d_train_label)
```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut

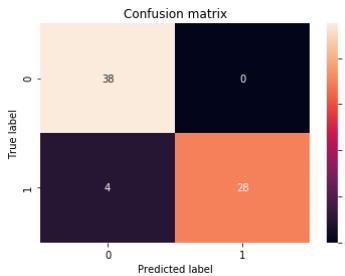
```
In [151]: from sklearn import tree
.....
Out[151]:
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_weight_fraction_leaf=0.0, presort=False,
random_state=None, splitter='best')
```

Gambar 4.121 Soal 5

6. Soal 6

```
1 from sklearn.ensemble import RandomForestClassifier
2 clf = RandomForestClassifier(n_estimators=80)
3 clf.fit(d_train_att, d_train_label)
4 clf.predict(d_test_att)
5
6 #%%soal 6
7
8 from sklearn.metrics import confusion_matrix
9 pred_labels=clf.predict(d_test_att)
10 cm=confusion_matrix(d_test_label, pred_labels)
11
12 import matplotlib.pyplot as plt
13 import seaborn as sns
14 sns.heatmap(cm, annot=True)
15 plt.title('Confusion matrix')
16 plt.ylabel('True label')
17 plt.xlabel('Predicted label')
18 plt.show()
```

Plot confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut



Gambar 4.122 Soal 6

7. Soal 7

```

1 from sklearn.model_selection import cross_val_score
2 scores=cross_val_score(clf,d_train_att,d_train_label, cv=5)
3 skor_rata2=scores.mean()
4 skoresd=scores.std()
5 print("Accuracy: %0.2f (+/- %0.2f)" % (skor_rata2, skoresd * 2))

```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut

```

In [154]: from sklearn.model_selection import cross_val_score
...: scores=cross_val_score(clf,d_train_att,d_train_label, cv=5)
...: skor_rata2=scores.mean()
...: skoresd=scores.std()
...: print("Accuracy: %0.2f (+/- %0.2f)" % (skor_rata2, skoresd * 2))
Accuracy: 0.95 (+/- 0.05)

```

Gambar 4.123 Soal 7

8. Soal 8

```

1 import numpy as np
2 max_features_opts = range(5, 50, 5) #max_features_opts
3 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
4 rf_params = np.empty((len(max_features_opts)*len(
5     n_estimators_opts),4), float) #rf_params sebagai variabel
6     untuk menjumlahkan yang sudah di tentukan sebelumnya
7 i = 0
8 for max_features in max_features_opts: #pengulangan
9     for n_estimators in n_estimators_opts: #pengulangan
10         clf = RandomForestClassifier(max_features=
11             max_features, n_estimators=n_estimators) #menampilkan
12             variabel csf
13             scores = cross_val_score(clf, d_train_att,
14             d_train_label, cv=5) #scores sebagai variabel training
15             rf_params[i,0] = max_features #index 0
16             rf_params[i,1] = n_estimators #index 1
17             rf_params[i,2] = scores.mean() #index 2

```

```

13     rf_params[i,3] = scores.std() * 2 #index 3
14     i += 1 #dengan ketentuan i += 1
15     print("Max features: %d, num estimators: %d, accuracy
16       : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
mean(), scores.std() * 2))
#print hasil pengulangan yang sudah ditentukan

```

Program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut

```

Max features: 40, num estimators: 90, accuracy: 0.95 (+/- 0.05)
Max features: 40, num estimators: 110, accuracy: 0.94 (+/- 0.06)
Max features: 40, num estimators: 130, accuracy: 0.94 (+/- 0.06)
Max features: 40, num estimators: 150, accuracy: 0.94 (+/- 0.06)
Max features: 40, num estimators: 170, accuracy: 0.94 (+/- 0.06)
Max features: 40, num estimators: 190, accuracy: 0.94 (+/- 0.07)
Max features: 45, num estimators: 10, accuracy: 0.91 (+/- 0.08)
Max features: 45, num estimators: 30, accuracy: 0.92 (+/- 0.04)
Max features: 45, num estimators: 50, accuracy: 0.94 (+/- 0.07)
Max features: 45, num estimators: 70, accuracy: 0.93 (+/- 0.07)
Max features: 45, num estimators: 90, accuracy: 0.93 (+/- 0.06)
Max features: 45, num estimators: 110, accuracy: 0.94 (+/- 0.06)
Max features: 45, num estimators: 130, accuracy: 0.94 (+/- 0.05)
Max features: 45, num estimators: 150, accuracy: 0.94 (+/- 0.05)
Max features: 45, num estimators: 170, accuracy: 0.94 (+/- 0.06)
Max features: 45, num estimators: 190, accuracy: 0.94 (+/- 0.06)

```

Gambar 4.124 Soal 8

4.9.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundException: [Errno 2] File b'csv_ariq.csv' does not exist: b'csv_ariq.csv'
```

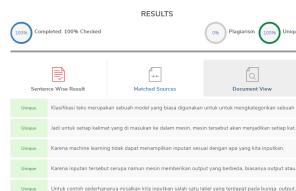
Gambar 4.125 FileNotFoundError

2. Cara Penangan Error

- **FileNotFoundException**

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.9.4 Bukti Tidak Plagiat



Gambar 4.126 Bukti Tidak Melakukan Plagiat Chapter 4

4.9.5 Link Youtube

<https://youtu.be/lLxUddy2kW4>

4.10 1174086 - Tia Nur candida

4.10.1 Teori

1. Klasifikasi Text

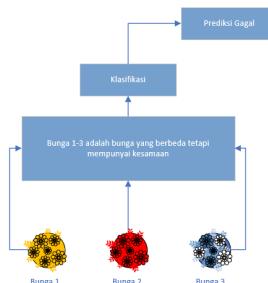
Klasifikasi teks merupakan cara dalam memilah milah data teks berdasarkan parameter tertentu dengan data yang bersifat dokumen ataupun teks yang memiliki kumpulan teks di dalamnya, serta teks itu sendiri bertipe data char atau string yang mudah untuk diolah.



Gambar 4.127 Klasifikasi Text

2. Mengapa Klasifikasi Bunga tidak dapat menggunakan Machine Learning

Karena klasifikasinya menggunakan tipe data yang dimana attributanya memiliki nilai data berupa vektor dengan perbandingan masing - masing data yang dimiliki memiliki sedikit perbedaan, sehingga program atau sistem tidak dapat membedakan dengan tepat antara gambar 1 dan gambar 2 dikarenakan memiliki perbedaan yang hampir tidak dapat dilihat pada beberapa contoh gambar. Untuk ilustrasi dapat dilihat pada gambar



Gambar 4.128 Klasifikasi Bunga

3. Teknik Pembelajaran machine learning pada teks kata-kata di youtube
Contohnya pada saat kita membuka sebuah video maka di sebelah kanan ada list "berikutnya", pada saat itu mesin melakukan ujicoba dan apabila anda menekan salah satu dari video tersebut maka hal tersebut akan direkam dan disimpan oleh mesin tersebut dan kemudian akan memutar yang ada pada list selanjutnya.



Gambar 4.129 Machine Learning Youtube

4. Jelaskan apa yang dimaksud vektorisasi data.

Pemecahan data menjadi bagian-bagian yang lebih sederhana contoh ada sebuah paragraf, dari paragraf tersebut akan dibagi-bagi menjadi kalimat, yang nantinya akan dibagi-bagi kembali menjadi perkata.

5. Jelaskan apa itu bag of words dengan kata-kata yang sederhana dan ilustrasi sendiri

5. Model bag-of-words adalah representasi penyederhanaan yang digunakan dalam pemrosesan bahasa alami dan pengambilan informasi (IR). Dalam model ini, sebuah teks (seperti kalimat atau dokumen) direpresentasikan sebagai tas (multiset) dari kata-katanya, mengabaikan tata bahasa dan bahkan urutan kata tetapi menjaga multiplisitas. Model bag-of-words juga telah digunakan untuk visi komputer. Model bag-of-words umumnya digunakan dalam metode klasifikasi dokumen di mana (frekuensi) kemunculan setiap kata digunakan sebagai fitur untuk melatih classifier

| | Document 1 | Document 2 |
|-------|------------|------------|
| Term | Document 1 | Document 2 |
| aid | 0 1 | |
| all | 1 0 | |
| back | 1 0 | |
| brown | 1 0 | |
| come | 1 0 | |
| good | 1 0 | |
| fox | 1 0 | |
| men | 1 0 | |
| lazy | 1 0 | |
| the | 1 0 | |
| time | 0 1 | |
| to | | 1 0 |
| of | | 1 0 |
| for | | 1 0 |
| party | | 1 0 |
| their | | 1 0 |
| over | | 1 0 |
| had | | 1 0 |

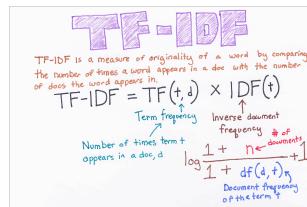
Stopword List

Gambar 4.130 Bag of Words

6. Jelaskan apa itu TF-IDF, ilustrasikan dengan gambar sendiri.

metode algoritma yang berguna untuk menghitung bobot setiap kata yang umum digunakan. Metode ini juga terkenal efisien, mudah dan

memiliki hasil yang akurat. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Secara sederhana, metode TF-IDF digunakan untuk mengetahui berapa sering suatu kata muncul di dalam dokumen.



Gambar 4.131 TF-IDF

4.10.2 Praktek

1. import data pandas dan 500 baris data dumy kemudian di jelaskan tiap barisnya.

```
1 import pandas as pd #mengimport library pandas dan menamainya
pd
2 #%%
3 tia = pd.read_csv("D:/TI/SMT 6/AI/Chapter4/New folder/src
    /1174086.csv") #membuat variable bernama tia dan
    mengisinya dengan data dari dataset dummy yang telah
    dibuat
4 a = tia.head() #untuk melihat 5 baris pertama dari data tia
5 tia.shape #untuk mengetahui berapa banyak baris data
6 print(a) #menampilkan isi dari varibale a pada console
```

The screenshot shows the Jupyter Notebook interface with the following content:

```
In [3]: tia = pd.read_csv("D:/TI/SMT 6/AI/Chapter4/New folder/src/1174086.csv")
Out[3]: DataFrame with 500 rows and 5 columns:
         id first_name last_name email   gender
0       1      Angie     Bentje  abentje@wpisqido.com  female
1       2      Karen      Tokan  stokan@wpisqido.de  female
2       3      Odile      gertegos@wpisqido.de  female
3       4      Callie     Riley  cpilley@wpisqido.com  female
4       5      Callie     Riley  cpilley@wpisqido.com  female
```

Gambar 4.132 Data Dummy

2. memecah data prame menjadi dua yag pertama 450 dan kedua sisanya

```
1 #%%
```

```

2 dtra = tia[:450] #memasukkan 450 data pertama ke dalam
   variable dtra
3 dtes = tia[450:] #memasukkan 50 data terakhir kedalam
   variable dtes

```

```

dtes DataFrame (50, 5) [Column names: id, first_name, last_name, email, gender]
dtra DataFrame (450, 5) [Column names: id, first_name, last_name, email, gender]

```

Gambar 4.133 Pisah Data

3. praktik vektorisasi

```

1 #%% memasukkan data dari file csv tersebut ke dalam variable
   data
2 data=pd.read_csv("D:/TI/SMT 6/AI/Chapter4/New folder/src
   /1174086.csv")
3 spam=data.query('CLASS == 1')
4 nospam=data.query('CLASS == 0')
5 #%% melakukan fungsi bag of word dengan cara menghitung semua
   kata
6 from sklearn.feature_extraction.text import CountVectorizer
7 vectorizer = CountVectorizer()
8 #%% melakukan bag of word pada dataframe pada colom CONTENT
9 data_vektorisasi = vectorizer.fit_transform(data['CONTENT'])
10 #%% melihat isi vektorisasi
11 data_vektorisasi
12 #%% melihat isi data pada baris ke 345
13 print(data['CONTENT'][345])
14 #%% untuk mengambil apa saja nama kolom yang tersedia
15 dk=vectorizer.get_feature_names()
16 #%%: melakukan randomisasi agar hasil sempurna pada saat
   klasifikasi
17 dshuf = data.sample(frac=1)
18 #%%: membuat data traning dan testing
19 dk_train=dshuf[:300]
20 dk_test=dshuf[300:]
21 #%%: melakukan training pada data training dan di vektorisasi
22 dk_train_att=vectorizer.fit_transform(dk_train['CONTENT'])
23 print(dk_train_att)
24 #%% melakukan testing pada data testing dan di vektorisasi
25 dk_test_att=vectorizer.transform(dk_test['CONTENT'])
26 print(dk_test_att)
27 #%%: Dimana akan mengambil label spam dan bukan spam
28 dk_train_label=dk_train['CLASS']
29 print(dk_train_label)
30 dk_test_label=dk_test['CLASS']
31 print(dk_test_label)

```

lakukan import library pandas yang diinisialisasi menjadi pd setelah itu ada dibuat variabel data dengan method read_csv untuk membaca file berekstensikan csv yang dimasukan alamatnya pada kurung, lakukan klasifikasi atau pemilihan komentar yang berisi spam atau bukan spam

dengan parameter class samadengan 1 merupakan spam dan class samadengan 0 bukan spam setelah itu masukan librari CountVektorizer yang digunakan untuk vektorisasi data kemudian dilanjutkan pada bagian In[103] dibuat variabel yang berisi vektorisasi dari data pada data di field content setelah itu variabel tersebut di running hasilnya menunjukan 350 baris di kali 1738 kolom selanjutnya dicoba untuk memunculkan isi record pada baris ke 345 maka akan muncul isian dari baris tersebut. selanjutnya dibuat variabel dk atau daftar yang berisi data hasil vektorisasi setelah yang terdiri dari variabel dshuf yang berisi data komen yang di dalamnya di buat random yang nantinya akan dibut data training dan data testing dengan ketentuan data training 300 dan data testing sebanyak 50 setelah itu data training di lakukan vektorisasi dan data testing juga dilakukan vektorisasi setelah itu kedua data training dan testing tersebut dibuat label dengan parameter field CLASS pada tabel.

```
...: dk_train,dk_test,dk_train['CLASS']
...: print(dk_train_label)
...: dk_test_label=dk_test['CLASS']
...: print(dk_test_label)
who is going to reach the billion first : katy or taylor ?
#> [1] "katy"
#> [2] "taylor"
#> [3] "katy"
#> [4] "taylor"
#> [5] "katy"
#> [6] "taylor"
#> [7] "taylor"
#> [8] "taylor"
#> [9] "taylor"
#> [10] "taylor"
#> [11] "taylor"
#> [12] "taylor"
#> [13] "taylor"
#> [14] "taylor"
#> [15] "taylor"
#> [16] "taylor"
#> [17] "taylor"
#> [18] "taylor"
#> [19] "taylor"
#> [20] "taylor"
#> [21] "taylor"
#> [22] "taylor"
#> [23] "taylor"
#> [24] "taylor"
#> [25] "taylor"
#> [26] "taylor"
#> [27] "taylor"
#> [28] "taylor"
#> [29] "taylor"
#> [30] "taylor"
#> [31] "taylor"
#> [32] "taylor"
#> [33] "taylor"
#> [34] "taylor"
#> [35] "taylor"
#> [36] "taylor"
#> [37] "taylor"
#> [38] "taylor"
#> [39] "taylor"
#> [40] "taylor"
#> [41] "taylor"
#> [42] "taylor"
#> [43] "taylor"
#> [44] "taylor"
#> [45] "taylor"
#> [46] "taylor"
#> [47] "taylor"
#> [48] "taylor"
#> [49] "taylor"
#> [50] "taylor"
#> [51] "taylor"
#> [52] "taylor"
#> [53] "taylor"
#> [54] "taylor"
#> [55] "taylor"
#> [56] "taylor"
#> [57] "taylor"
#> [58] "taylor"
#> [59] "taylor"
#> [60] "taylor"
#> [61] "taylor"
#> [62] "taylor"
#> [63] "taylor"
#> [64] "taylor"
#> [65] "taylor"
#> [66] "taylor"
#> [67] "taylor"
#> [68] "taylor"
#> [69] "taylor"
#> [70] "taylor"
#> [71] "taylor"
#> [72] "taylor"
#> [73] "taylor"
#> [74] "taylor"
#> [75] "taylor"
#> [76] "taylor"
#> [77] "taylor"
#> [78] "taylor"
#> [79] "taylor"
#> [80] "taylor"
#> [81] "taylor"
#> [82] "taylor"
#> [83] "taylor"
#> [84] "taylor"
#> [85] "taylor"
#> [86] "taylor"
#> [87] "taylor"
#> [88] "taylor"
#> [89] "taylor"
#> [90] "taylor"
#> [91] "taylor"
#> [92] "taylor"
#> [93] "taylor"
#> [94] "taylor"
#> [95] "taylor"
#> [96] "taylor"
#> [97] "taylor"
#> [98] "taylor"
#> [99] "taylor"
#> [100] "taylor"
#> [101] "taylor"
#> [102] "taylor"
#> [103] "taylor"
#> [104] "taylor"
#> [105] "taylor"
#> [106] "taylor"
#> [107] "taylor"
#> [108] "taylor"
#> [109] "taylor"
#> [110] "taylor"
#> [111] "taylor"
#> [112] "taylor"
#> [113] "taylor"
#> [114] "taylor"
#> [115] "taylor"
#> [116] "taylor"
#> [117] "taylor"
#> [118] "taylor"
#> [119] "taylor"
#> [120] "taylor"
#> [121] "taylor"
#> [122] "taylor"
#> [123] "taylor"
#> [124] "taylor"
#> [125] "taylor"
#> [126] "taylor"
#> [127] "taylor"
#> [128] "taylor"
#> [129] "taylor"
#> [130] "taylor"
#> [131] "taylor"
#> [132] "taylor"
#> [133] "taylor"
#> [134] "taylor"
#> [135] "taylor"
#> [136] "taylor"
#> [137] "taylor"
#> [138] "taylor"
#> [139] "taylor"
#> [140] "taylor"
#> [141] "taylor"
#> [142] "taylor"
#> [143] "taylor"
#> [144] "taylor"
#> [145] "taylor"
#> [146] "taylor"
#> [147] "taylor"
#> [148] "taylor"
#> [149] "taylor"
#> [150] "taylor"
#> [151] "taylor"
#> [152] "taylor"
#> [153] "taylor"
#> [154] "taylor"
#> [155] "taylor"
#> [156] "taylor"
#> [157] "taylor"
#> [158] "taylor"
#> [159] "taylor"
#> [160] "taylor"
#> [161] "taylor"
#> [162] "taylor"
#> [163] "taylor"
#> [164] "taylor"
#> [165] "taylor"
#> [166] "taylor"
#> [167] "taylor"
#> [168] "taylor"
#> [169] "taylor"
#> [170] "taylor"
#> [171] "taylor"
#> [172] "taylor"
#> [173] "taylor"
#> [174] "taylor"
#> [175] "taylor"
#> [176] "taylor"
#> [177] "taylor"
#> [178] "taylor"
#> [179] "taylor"
#> [180] "taylor"
#> [181] "taylor"
#> [182] "taylor"
#> [183] "taylor"
#> [184] "taylor"
#> [185] "taylor"
#> [186] "taylor"
#> [187] "taylor"
#> [188] "taylor"
#> [189] "taylor"
#> [190] "taylor"
#> [191] "taylor"
#> [192] "taylor"
#> [193] "taylor"
#> [194] "taylor"
#> [195] "taylor"
#> [196] "taylor"
#> [197] "taylor"
#> [198] "taylor"
#> [199] "taylor"
#> [200] "taylor"
#> [201] "taylor"
#> [202] "taylor"
#> [203] "taylor"
#> [204] "taylor"
#> [205] "taylor"
#> [206] "taylor"
#> [207] "taylor"
#> [208] "taylor"
#> [209] "taylor"
#> [210] "taylor"
#> [211] "taylor"
#> [212] "taylor"
#> [213] "taylor"
#> [214] "taylor"
#> [215] "taylor"
#> [216] "taylor"
#> [217] "taylor"
#> [218] "taylor"
#> [219] "taylor"
#> [220] "taylor"
#> [221] "taylor"
#> [222] "taylor"
#> [223] "taylor"
#> [224] "taylor"
#> [225] "taylor"
#> [226] "taylor"
#> [227] "taylor"
#> [228] "taylor"
#> [229] "taylor"
#> [230] "taylor"
#> [231] "taylor"
#> [232] "taylor"
#> [233] "taylor"
#> [234] "taylor"
#> [235] "taylor"
#> [236] "taylor"
#> [237] "taylor"
#> [238] "taylor"
#> [239] "taylor"
#> [240] "taylor"
#> [241] "taylor"
#> [242] "taylor"
#> [243] "taylor"
#> [244] "taylor"
#> [245] "taylor"
#> [246] "taylor"
#> [247] "taylor"
#> [248] "taylor"
#> [249] "taylor"
#> [250] "taylor"
#> [251] "taylor"
#> [252] "taylor"
#> [253] "taylor"
#> [254] "taylor"
#> [255] "taylor"
#> [256] "taylor"
#> [257] "taylor"
#> [258] "taylor"
#> [259] "taylor"
#> [260] "taylor"
#> [261] "taylor"
#> [262] "taylor"
#> [263] "taylor"
#> [264] "taylor"
#> [265] "taylor"
#> [266] "taylor"
#> [267] "taylor"
#> [268] "taylor"
#> [269] "taylor"
#> [270] "taylor"
#> [271] "taylor"
#> [272] "taylor"
#> [273] "taylor"
#> [274] "taylor"
#> [275] "taylor"
#> [276] "taylor"
#> [277] "taylor"
#> [278] "taylor"
#> [279] "taylor"
#> [280] "taylor"
#> [281] "taylor"
#> [282] "taylor"
#> [283] "taylor"
#> [284] "taylor"
#> [285] "taylor"
#> [286] "taylor"
#> [287] "taylor"
#> [288] "taylor"
#> [289] "taylor"
#> [290] "taylor"
#> [291] "taylor"
#> [292] "taylor"
#> [293] "taylor"
#> [294] "taylor"
#> [295] "taylor"
#> [296] "taylor"
#> [297] "taylor"
#> [298] "taylor"
#> [299] "taylor"
#> [300] "taylor"
```

Gambar 4.134 Vektorisasi

4. klasifikasi SVM

```
1 from sklearn import svm
2 clfsvm = svm.SVC()
3 clfsvm . fit (dk_train_att , dk_train_label)
4 clfsvm . score (dk_test_att , dk_test_label)
```

import librari svm dari sklearn kemudian membuat variabel clfsvm berisikan method svc setelah itu variabel tersebut di berikan method fit dengan isian data train vektorisasi dan data training label yang berguna untuk melatih data tersebut setelah itu di coba untuk memunculkan score atau akurasi dari data tersebut menggunakan data testing vektorisasi dan data testing label.

```
In [66]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(dk_train_att, dk_train_label)
...: clfsvm.score(dk_test_att, dk_test_label)
Out[66]: 0.94
```

Gambar 4.135 SVM

5. klasifikasi decision tree

```
1 from sklearn import tree  
2 clftree = tree.DecisionTreeClassifier()  
3 clftree.fit(dk_train_att, dk_train_label)  
4 clftree.score(dk_test_att, dk_test_label)
```

import librari tree dari sklearn kemudian membuat variabel clftree berisikan method DecisionTreeClasifier setelah itu variabel tersebut di berikan method fit dengan isian data train vektorisasi dan data training label yang berguna untuk melatih data tersebut agar dapat digunakan pada codingan selanjutnya setelah itu di coba untuk memunculkan score atau akurasi dari data tersebut menggunakan data testing vektorisasi dan data testing label.

```
In [67]: from sklearn import tree
      ...: clftree = tree.DecisionTreeClassifier()
      ...: clftree.fit(dk_train_att, dk_train_label)
      ...: clftree.score(dk_test_att, dk_test_label)
Out[67]: 0.96
```

Gambar 4.136 Desicion Tree

6. plot comfusion matrix

```
1 from sklearn.metrics import confusion_matrix  
2 pred_labels = clftree.predict(dk_test_att)  
3 cm = confusion_matrix(dk_test_label, pred_labels)  
4 cm
```

import library confusion matrix selanjutnya dilakukan prediksi pada pada data tes nya kemudian data tersebut di masukan kedalam variabel cm dengan method confusion matrix yang di dalamnya terdapat data dari variabel perd label dan dk test label setelah itu variabel cm tersebut di running maka akan memunculkan nilai matrixnya.

```
In [80]: from sklearn.metrics import confusion_matrix
...: pred_labels = clftree.predict(dk_test_att)
...: cm = confusion_matrix(dk_test_label, pred_labels)
...: cm
Out[80]:
array([[25,  1],
       [ 1, 23]], dtype=int64)
```

Gambar 4.137 Confusion Matrix

7. cross validation

```
1 #%%  
2 from sklearn.model_selection import cross_val_score  
3 scores = cross_val_score(clftree, dk_train_att, dk_train_label,  
    cv=5)  
4 scorerata2=scores.mean()
```

```

5 scorersd=scores . std ()
6 #%%:
7 from sklearn.model_selection import cross_val_score
8 scores = cross_val_score(clftree , dk_train_att ,
9 dk_train_label , cv=5)
# show average score and +/- two standard deviations away (
10 # covering 95
11 #%% of scores)
12 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean() ,
13 scores.std() * 2))
14 #%%:
15 scorestree = cross_val_score(clftree , dk_train_att ,
16 dk_train_label , cv=5)
17 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean() ,
18 scorestree.std() * 2))
19 #%%:
20 scoressvm = cross_val_score(clfsvm , dk_train_att ,
21 dk_train_label , cv=5)
22 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean() ,
23 scoressvm.std() * 2))

```

memunculkan nilai akurasi dari tiga metode yaitu random forest, decision tree, dan klasifikasi svm (suport vector machine) diamana akan di bandingkan tingkat akurasi dari semua hasil akurasi mana yang terbaik dan lebih akurat pada hasilnya data yang paling akurat yaitu random forest.

```

In [82]: from sklearn.model_selection import cross_val_score
...: ...: scores = cross_val_score(clftree , dk_train_att , dk_train_label , cv=5)
...: ...: scorestree = scores.mean()
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean() ,
...: ...: scores.std() * 2))
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean() ,
...: ...: scorestree.std() * 2))
Accuracy: 0.82 (+/- 0.07)

In [83]: from sklearn.model_selection import cross_val_score
...: ...: scores = cross_val_score(clftree , dk_train_att , dk_train_label , cv=5)
...: ...: # show average score and +/- two standard deviations away (covering 95
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean() ,
...: ...: scores.std() * 2))
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean() ,
...: ...: scorestree.std() * 2))
Accuracy: 0.82 (+/- 0.07)

In [84]: scorestree = cross_val_score(clftree , dk_train_att , dk_train_label , cv=5)
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean() ,
...: ...: scorestree.std() * 2))
Accuracy: 0.82 (+/- 0.07)

In [85]: scoressvm = cross_val_score(clfsvm , dk_train_att , dk_train_label , cv=5)
...: ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean() ,
...: ...: scoressvm.std() * 2))
Accuracy: 0.91 (+/- 0.08)

```

Gambar 4.138 Cross Validation

8. Pengamatan program

```

1 #%%
2 import numpy as np
3 max_features_opts = range(1, 10, 1)
4 n_estimators_opts = range(2, 40, 4)
5 rf_params = np.empty((len(max_features_opts)*len(
6 n_estimators_opts),4) , float)
7 i = 0
8 for max_features in max_features_opts:
9     for n_estimators in n_estimators_opts:
10         clf = RandomForestClassifier(max_features=
11             max_features , n_estimators=n_estimators)
12         scores = cross_val_score(clf , dk_train_att ,
13 dk_train_label , cv=5)
14         rf_params[i,0] = max_features
15         rf_params[i,1] = n_estimators

```

```

13     rf_params[i,2] = scores.mean()
14     rf_params[i,3] = scores.std() * 2
15     i += 1
16     print("Max features: %d, num estimators: %d, accuracy
17 : %0.2f (+/- %0.2f)" %
18      (max_features, n_estimators, scores.mean(), scores.std() *
19       2))

```

terdapat grafik data yang terdapat dari grafik tersebut di dapat dari codingan dengan cara pengulangan data masing masing 10 kali setelah itu di eksekusi menjadi grafik berbentuk 3D pada gambar tersebut menunjukan rasio dari yang terrendah yaitu data SVM kemudian data decision tree dan hasil random forest.

```

#n_estimators=100, max_features='auto', max_depth=None, min_samples_split=2,
n_estimators=n_estimators, estimator=rf)
...:
scores = cross_val_score(rf, X_train_att, dk_train_label, cv=cv)
...:
rf_params[:,0] = max_features
...:
rf_params[:,1] = n_estimators
rf_params[:,2] = scores.mean()
...:
rf_params[:,3] = scores.std() * 2
...:
for i in range(10):
...:
    print("Max features: %d, num estimators: %d, accuracy: %0.2f (+/-
...:
    0.2f)" %
...:
    (max_features, n_estimators, scores.mean(), scores.std() * 2))
...:
Max features: 1, num estimators: 2, accuracy: 0.73 (+/- 0.06)
Max features: 1, num estimators: 6, accuracy: 0.88 (+/- 0.00)
Max features: 1, num estimators: 10, accuracy: 0.89 (+/- 0.00)
Max features: 1, num estimators: 14, accuracy: 0.88 (+/- 0.12)
Max features: 1, num estimators: 18, accuracy: 0.89 (+/- 0.04)
Max features: 1, num estimators: 22, accuracy: 0.87 (+/- 0.07)
Max features: 3, num estimators: 26, accuracy: 0.88 (+/- 0.05)
Max features: 3, num estimators: 30, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 34, accuracy: 0.89 (+/- 0.08)
Max features: 3, num estimators: 38, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 42, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 46, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 50, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 54, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 58, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 62, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 66, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 70, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 74, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 78, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 82, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 86, accuracy: 0.89 (+/- 0.05)
Max features: 5, num estimators: 90, accuracy: 0.89 (+/- 0.05)

```

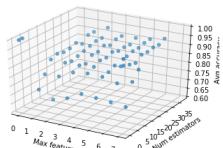
Gambar 4.139 Pengamatan Program

Berikut adalah untuk grafik

```

1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 from matplotlib import cm
4 fig = plt.figure()
5 fig.clf()
6 ax = fig.gca(projection='3d')
7 x = rf_params[:,0]
8 y = rf_params[:,1]
9 z = rf_params[:,2]
10 ax.scatter(x, y, z)
11 ax.set_zlim(0.6, 1)
12 ax.set_xlabel('Max features')
13 ax.set_ylabel('Num estimators')
14 ax.set_zlabel('Avg accuracy')
15 plt.show()

```



Gambar 4.140 Grafik

4.10.3 Penanganan Error

4.10.3.1 Screenshoot Error

1. Error 1

```
r1_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
NameError: name 'np' is not defined
```

Gambar 4.141 Error1

2. Error 2

```
file "pandas\_libs\hashtable_class_helper.pxi", line 1492, in
pandas\_libs\hashtable\PyObjectHashTable.get_item
file "pandas\_libs\hashtable_class_helper.pxi", line 1500, in
pandas\_libs\hashtable\PyObjectHashTable.get_item
KeyError: 'CONTENTz'
```

Gambar 4.142 Error2

3. Error 3

```
file "pandas\_libs\parsers.pyx", line 695, in
pandas\_libs\parsers\TextHeader._setup_parser_source
FileNotFoundError: File b'D:\SemesterV\ATI\Chapter40d0b0.csv' does not exist
```

Gambar 4.143 Error3

4. Error 4

```
file "pandas\_libs\hashtable_class_helper.pxi", line 964, in
pandas\_libs\hashtable\Int64HashTable.get_item
KeyError: 5000
```

Gambar 4.144 Error4

4.10.3.2 Kode Error dan Jenisnya

1. Kode Error 1 jenis Name Error

```
max_features_opts = np.empty(39, 1)
n_estimators_opts = np.empty(40, 4)
r1_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
i = 0
```

Gambar 4.145 Error1

2. Kode Error 2 jenis Key Error

```
data_vektorisasi = vectorizer.fit_transform(data['CONTENTz'])
```

Gambar 4.146 Error2

3. Kode Error 3 jenis FileNotFoundError

```
#%%
tia = pd.read_csv("D:/TI/SMT 6/AI/Chapter4/New folder/src/1174886.csv")
a = tia.head() #untuk melihat 5 baris pertama dari data tia
tia.shape #untuk mengetahui berapa banyak baris data
print(a) #menampilkan isi dari variabel a pada console
```

Gambar 4.147 Error3

4. Kode Error 4 Key Error

```
#%%
print(data['CONTENT'][5000])
```

Gambar 4.148 Error4

4.10.3.3 Solusi

1. Mengimport library numpy sebagai np

```
import numpy as np
max_features_opts = range(1, 10, 1)
n_estimators_opts = range(2, 40, 4)
rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
```

Gambar 4.149 Solusi 1

2. Mengganti CONTENTz menjadi CONTENT

```
data_vektorisasi = vectorizer.fit_transform(data['CONTENT'])
```

Gambar 4.150 Solusi 2

3. Merubah backslash menjadi slash biasa

```
#%%
tia = tia.head() #untuk melihat 5 baris pertama dari data tia
tia.shape #untuk mengetahui berapa banyak baris data
print(a) #menampilkan isi dari variabel a pada console
```

Gambar 4.151 Solusi 3

4. Merubah data menjadi dibawah 350

```
#%%
print(data['CONTENT'][349])
```

Gambar 4.152 Solusi 4

4.10.4 Link Youtube

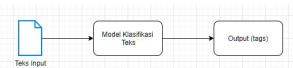
<https://youtu.be/Zsk8IXYg7Gk>

4.11 1174071 - Muhammad Abdul Gani Wijaya

4.11.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.153 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri.

Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.154 Klasifikasi Bunga.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.

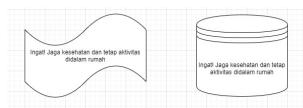
Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.155 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.
Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.
5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.156 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.157 TF-IDF.

4.11.2 Praktek Program

1. Soal 1

```

1 #%% Soal 1
2 import pandas as pd
3 #digunakan untuk mengimport library pandas dengan alias pd

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :

Gambar 4.158 Hasil Soal 1.

2. Soal 2

```

1 #membaca file csv
2
3 #%% Soal 2

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :

Gambar 4.159 Hasil Soal 2.

3. Soal 3

```

1 #untuk membagi data training menjadi 450
2 d_test=pd[450:]
3 #untuk membagi data menjadi 50 atau sisa dari data yang tersedia
4
5 #%% Soal 3
6 import pandas as gani
7 #untuk import library pandas berguna untuk mengelola dataframe
8 gani = gani.read_csv("C:/Users/muham/Downloads/Compressed/KB3C-master/KB3C-master/src/1174071/4/Youtube03-Shakira.csv")

```

```

9 #untukmembaca file dengan format csv
10
11 spam=gani.query( 'CLASS == 1' )
12 #untuk membagi tabel spam
13 nospam=gani.query( 'CLASS == 0' )
14 #untuk membagi tabel no spam
15
16 from sklearn.feature_extraction.text import CountVectorizer
17 #import countvectorizer berfungsi untuk memecah data tersebut
18     menjadi sebuah kata yang lebih sederhana
19 vectorizer = CountVectorizer ()
20 #menjalankan fungsi tersebut , pada code ini tidak ada
21     hasilnya dikarenakan spyder tidak mendukung hasil dari
22     instasiasi .
23
24 dvec = vectorizer.fit_transform(gani[ 'CONTENT' ])
25 #melakukan pemecahan data pada dataframe yang terdapat pada
26     kolom konten
27 dvec #menampilkan hasil dari code sebelumnya
28
29 Daptarkata= vectorizer.get_feature_names()

```

Dengan menggunakan $1174069 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

| ferry - DataFrame | | | | | |
|-------------------|---------------|--------------|-----------------|--------------------|-------|
| Index | COMMENT_ID | AUTHOR | DATE | CONTENT | CLASS |
| 0 | r1234567890.. | Cory Wilson | 2015-05-28T21.. | re | 0 |
| 1 | r1234567890.. | Iain Gating | 2015-05-28T21.. | re | 0 |
| 2 | r1234567890.. | Led Public | 2015-05-28T21.. | I'm trying to ha.. | 1 |
| 3 | r1234567890.. | Cheryl Fox | 2015-05-28T21.. | Rock...lol.. | 0 |
| 4 | r1234567890.. | PATICK_TW | 2015-05-28T21.. | Party rock | 0 |
| 5 | r1234567890.. | Brian Bral | 2015-05-28T0.. | Shuffle | 0 |
| 6 | r1234567890.. | Brian Bral | 2015-05-28T0.. | Omg | 0 |
| 7 | r1234567890.. | Alex Derev | 2015-05-28T0.. | Just really .. | 0 |
| 8 | r1234567890.. | Simeon | 2015-05-28T0.. | Assassinate /> | 0 |
| 9 | r1234567890.. | Silvia Bauer | 2015-05-28T0.. | Incredible so .. | 0 |
| 10 | r1234567890.. | Uniteide | 2015-05-28T0.. | I love you so much | 0 |
| 11 | r1234567890.. | gentle maria | 2015-05-28T0.. | THIS LIEEE | 0 |
| 12 | r1234567890.. | | 2015-05-28T0.. | I miss you so .. | 0 |
| 13 | r1234567890.. | Alex Jansen | 2015-05-28T0.. | OMG THE PRESS | 0 |
| 14 | r1234567890.. | Aliden Hill | 2015-05-27T2.. | Best song of .. | 0 |
| 15 | r1234567890.. | joe paulin | 2015-05-27T2.. | super nice, | 0 |
| 16 | r1234567890.. | silvia Bauer | 2015-05-27T2.. | so many | 0 |
| 17 | r1234567890.. | bilal kemo | 2015-05-27T2.. | unbelievable | 0 |
| 18 | r1234567890.. | SoulInADance | 2015-05-27T1.. | PARTY ROCK | 0 |
| 19 | r1234567890.. | Phuong Lin | 2015-05-27T1.. | Thums up if .. | 0 |
| 20 | r1234567890.. | Gontalo | 2015-05-27T1.. | THIS IS .. | 0 |
| | r1234567890.. | utahr70N | 2015-05-27T1.. | so cool!!!!!! | 0 |

Gambar 4.160 Hasil Soal 3.

4. Soal 4

```

1 dshuf = gani.sample( frac=1)
2
3 d_train=dshuf[:300]
4 d_test=dshuf[300:]

```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```
In [7]: from sklearn import svm
... clfsvm = svm.SVR(gamma = 'auto')
... clfsvm.fit(d_train_att, d_train_label)
Out[7]:
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1,
     gamma='auto',
     kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
     verbose=False)
```

Gambar 4.161 Hasil Soal 4.

5. Soal 5

```
1 d_train_att
2
3 d_train_label=d_train[ 'CLASS' ]
4 d_test_label=d_test[ 'CLASS' ]
```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut :

```
In [8]: from sklearn import tree
... clftree = tree.DecisionTreeClassifier()
... clftree.fit(d_train_att, d_train_label)
Out[8]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated', random_state=None, splitter='best')
```

Gambar 4.162 Hasil Soal 5.

6. Soal 6

```
1
2 from sklearn import svm
3 clfsvm = svm.SVR(gamma = 'auto')
4 clfsvm . fit (d_train_att , d_train_label)
5
6 #%%soal 5
7
8 from sklearn import tree
9 clftree = tree.DecisionTreeClassifier()
10 clftree . fit (d_train_att , d_train_label)
11
12 #%%soal 6/
13
14 from sklearn.metrics import confusion_matrix
15 pred_labels=clftree.predict(d_test)
16 cm=confusion_matrix(d_test_label , pred_labels)
17
18 #%%
19
20 import matplotlib.pyplot as plt
21
22 def plot_confusion_matrix(cm, classes,
```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...
... def plot_confusion_matrix(cm, classes,
...                          normalize=False,
...                          title='Confusion matrix',
...                          cmap=plt.cm.Blues):
...
...     if normalize:
...         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...         print("Normalized confusion matrix")
...     else:
...         print('Confusion matrix, without normalization')
...
...     print(cm)
...
...     plt.imshow(cm,
```

Gambar 4.163 Hasil Soal 6.

7. Soal 7

```
1             title='Confusion matrix' ,
2             cmap=plt.cm.Blues):
3
4     if normalize:
5         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
6         print("Normalized confusion matrix")
7     else:
8         print('Confusion matrix, without normalization')
```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```
In [12]: from sklearn.model_selection import cross_val_score
...
... scores=cross_val_score(clftree,d_train_att,d_train_label,cv=5)
...
... skor_rata2=scores.mean()
... skoresd=scores.std()
```

Gambar 4.164 Hasil Soal 7.

8. Soal 8

```
1     print(cm)
2
3 #%%soal 7
4
5 from sklearn.model_selection import cross_val_score
6
7 scores=cross_val_score(clftree ,d_train_att ,d_train_label ,cv
=5)
8
9 skor_rata2=scores .mean()
10 skoresd=scores .std()
11
12 #%%soal 8 /
```

```

13
14 max_features_opts = range(5, 50, 5)
15 #membuat max_features_opts sebagai variabel untuk membuat
16     range 5,50,5
17 n_estimators_opts = range(10, 200, 20)
18 #membuat n_estimators_opts sebagai variabel untuk membuat
19     range 10,200,20

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```

Max features: 25, num estimators: 100, accuracy: 0.46 (-/-: 0.03)
Max features: 30, num estimators: 100, accuracy: 0.33 (-/-: 0.02)
Max features: 35, num estimators: 100, accuracy: 0.43 (-/-: 0.02)
Max features: 39, num estimators: 50, accuracy: 0.43 (-/-: 0.03)
Max features: 39, num estimators: 70, accuracy: 0.44 (-/-: 0.03)
Max features: 39, num estimators: 100, accuracy: 0.44 (-/-: 0.03)
Max features: 39, num estimators: 130, accuracy: 0.45 (-/-: 0.03)
Max features: 39, num estimators: 150, accuracy: 0.45 (-/-: 0.03)
Max features: 39, num estimators: 170, accuracy: 0.45 (-/-: 0.04)
Max features: 39, num estimators: 180, accuracy: 0.46 (-/-: 0.03)
Max features: 39, num estimators: 190, accuracy: 0.46 (-/-: 0.03)
Max features: 39, num estimators: 200, accuracy: 0.46 (-/-: 0.03)
Max features: 35, num estimators: 100, accuracy: 0.34 (-/-: 0.03)
Max features: 35, num estimators: 130, accuracy: 0.43 (-/-: 0.02)
Max features: 35, num estimators: 50, accuracy: 0.43 (-/-: 0.02)
Max features: 35, num estimators: 70, accuracy: 0.44 (-/-: 0.03)
Max features: 35, num estimators: 100, accuracy: 0.44 (-/-: 0.03)
Max features: 35, num estimators: 130, accuracy: 0.45 (-/-: 0.03)
Max features: 35, num estimators: 150, accuracy: 0.46 (-/-: 0.03)
Max features: 35, num estimators: 170, accuracy: 0.46 (-/-: 0.03)
Max features: 35, num estimators: 180, accuracy: 0.46 (-/-: 0.03)
Max features: 35, num estimators: 200, accuracy: 0.46 (-/-: 0.03)
Max features: 40, num estimators: 10, accuracy: 0.34 (-/-: 0.02)
Max features: 40, num estimators: 30, accuracy: 0.43 (-/-: 0.02)
Max features: 40, num estimators: 50, accuracy: 0.43 (-/-: 0.03)
Max features: 40, num estimators: 70, accuracy: 0.44 (-/-: 0.03)
Max features: 40, num estimators: 100, accuracy: 0.44 (-/-: 0.03)
Max features: 40, num estimators: 130, accuracy: 0.45 (-/-: 0.03)
Max features: 40, num estimators: 150, accuracy: 0.46 (-/-: 0.03)
Max features: 40, num estimators: 170, accuracy: 0.46 (-/-: 0.03)
Max features: 40, num estimators: 180, accuracy: 0.46 (-/-: 0.03)
Max features: 40, num estimators: 200, accuracy: 0.46 (-/-: 0.03)
Max features: 45, num estimators: 10, accuracy: 0.34 (-/-: 0.02)
Max features: 45, num estimators: 30, accuracy: 0.43 (-/-: 0.02)
Max features: 45, num estimators: 50, accuracy: 0.43 (-/-: 0.03)
Max features: 45, num estimators: 70, accuracy: 0.44 (-/-: 0.03)
Max features: 45, num estimators: 100, accuracy: 0.44 (-/-: 0.03)
Max features: 45, num estimators: 130, accuracy: 0.45 (-/-: 0.03)
Max features: 45, num estimators: 150, accuracy: 0.46 (-/-: 0.03)
Max features: 45, num estimators: 170, accuracy: 0.46 (-/-: 0.03)
Max features: 45, num estimators: 180, accuracy: 0.46 (-/-: 0.03)
Max features: 45, num estimators: 200, accuracy: 0.46 (-/-: 0.03)

```

Gambar 4.165 Hasil Soal 8.

4.11.3 Penanganan Error

1. ScreenShoot Error

```

FileNotFoundException: [Error 2] File b'F://Semester 6//Artificial Intelligence/Tugas
4//src/fenny.csv" does not exist b'F://Semester 6//Artificial Intelligence/Tugas 4/
src/fenny.csv'

```

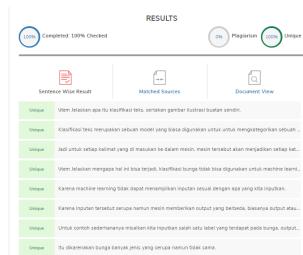
Gambar 4.166 SyntaxError

2. Cara Penangan Error

- **SyntaxError**

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.11.4 Bukti Tidak Plagiat



Gambar 4.167 Bukti Tidak Melakukan Plagiat Chapter 4

4.11.5 Link Youtube

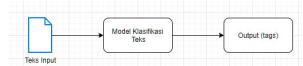
<https://youtu.be/X-xd9Nb78Gs>

4.12 1174071 - Muhammad Abdul Gani Wijaya

4.12.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.168 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri. Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga

banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.169 Klasifikasi Bunga.

3. Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.

Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



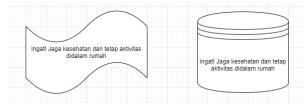
Gambar 4.170 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.

Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

5. Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.171 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada

suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

| | Doc 1 | Doc 2 | ... | Doc n |
|-----------|-------|-------|-----|-------|
| Term(s) 1 | 12 | 2 | ... | 1 |
| Term(s) 2 | 0 | 1 | ... | 0 |
| ... | ... | ... | ... | ... |
| Term(s) n | 0 | 6 | ... | 3 |

Gambar 4.172 TF-IDF.

4.12.2 Praktek Program

1. Soal 1

```
1 %% Soal 1
2 import pandas as pd
3 #digunakan untuk mengimport library pandas dengan alias pd
```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :

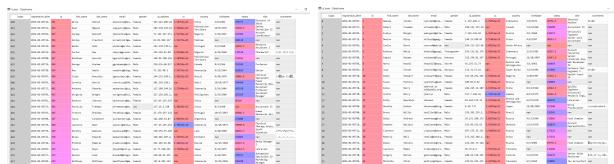


Gambar 4.173 Hasil Soal 1.

2. Soal 2

```
1 #membaca file csv
2
3 %% Soal 2
```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :



Gambar 4.174 Hasil Soal 2.

3. Soal 3

```

1 #untuk membagi data training menjadi 450
2 d_test=pd[450:]
3 #untuk membagi data menjadi 50 atau sisa dari data yang tersedia
4
5 %% Soal 3
6 import pandas as gani
7 #untuk import library pandas berguna untuk mengelola dataframe
8 gani = gani.read_csv("C:/Users/muham/Downloads/Compressed/
KB3C-master/KB3C-master/src/1174071/4/Youtube03-Shakira .
csv")
9 #untuk membaca file dengan format csv
10
11 spam=gani.query('CLASS == 1')
12 #untuk membagi tabel spam
13 nospam=gani.query('CLASS == 0')
14 #untuk membagi tabel no spam
15
16 from sklearn.feature_extraction.text import CountVectorizer
17 #import countvectorizer berfungsi untuk memecah data tersebut menjadi sebuah kata yang lebih sederhana
18 vectorizer = CountVectorizer()
19 #menjalankan fungsi tersebut, pada code ini tidak ada hasilnya dikarenakan spyder tidak mendukung hasil dari instasiasi.
20
21 dvec = vectorizer.fit_transform(gani['CONTENT'])
22 #melakukan pemecahan data pada dataframe yang terdapat pada kolom konten
23 dvec #menampilkan hasil dari code sebelumnya
24
25 Daptarkata= vectorizer.get_feature_names()

```

Dengan menggunakan $1174069 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

| Index | COMMENT_ID | AUTHOR | DATE | CONTENT | CLASS |
|-------|-------------------------------|-----------------|---|-------------------------|-------|
| 0 | 112610896d... Corey Wilson | 2015-05-28T2... | (a | hey guys, i'm trying to | 0 |
| 1 | 11247cc2a5d... Eric Geling | 2015-05-28T2... | elred but | 0 | |
| 2 | 11237c545fb... Lek Music | 2015-05-28T2... | hey guys, i have | 1 | |
| 3 | 11237c545fb... Chey Fox | 2015-05-28T2... | Part 2 of Rock...lol... | 0 | |
| 4 | 11237c545fb... PANTICO_7U | 2015-05-28T2... | Party Rock | 0 | |
| 5 | 11237c545fb... Brian Brad | 2015-05-28T2... | Shoutin | 0 | |
| 6 | 11237c545fb... Brian Brad | 2015-05-28T2... | 0 | 0 | |
| 7 | 11237c545fb... Alex Dafeo | 2015-05-28T2... | This song is | 0 | |
| 8 | 11237c545fb... Alex Dafeo | 2015-05-28T2... | just really... | 0 | |
| 9 | 11237c545fb... Giovanni | 2015-05-28T2... | awesomeness /> | 0 | |
| 10 | 11237c545fb... Silvia Basurto | 2015-05-28T2... | Incredible no. | 0 | |
| 11 | 11237c545fb... Unitewide | 2015-05-28T2... | song so much | 0 | |
| 12 | 11237c545fb... Rodriguez | 2015-05-28T2... | 2015 Lifef | 0 | |
| 13 | 11237c545fb... Alex Martin | 2015-05-28T2... | people dress, | 0 | |
| 14 | 11237c545fb... Alex Jensen | 2015-05-28T2... | last year of, | 0 | |
| 15 | 11237c545fb... Adam Hill | 2015-05-27T2... | Best song | 0 | |
| 16 | 11237c545fb... Joe pebb... | 2015-05-27T2... | super nice, | 0 | |
| 17 | 11237c545fb... Silvia Kewo | 2015-05-27T2... | super awesome | 0 | |
| 18 | 11237c545fb... SoulInTheDunes | 2015-05-27T2... | PARTY ROCK | 0 | |
| 19 | 11237c545fb... Phuong Linh | 2015-05-27T2... | Thumbs up if | 0 | |
| 20 | 11237c545fb... Gonzalez | 2015-05-27T2... | this is | 0 | |
| | 11237c545fb... Gonzalez | 2015-05-27T2... | presort='deprecated', random_state=None, splitter='best') | 0 | |

Gambar 4.175 Hasil Soal 3.

4. Soal 4

```

1 dshuf = gani.sample( frac=1)
2
3 d_train=dshuf[:300]
4 d_test=dshuf[300:]

```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```

In [7]: from sklearn import svm
... clfsvm = svm.SVC(gamma = 'auto')
... clfsvm.fit(d_train_att, d_train_label)
Out[7]:
SVC(C=1.0, cache_size=200, class_weight=None, degree=3, epsilon=0.1,
gamma='auto',
kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
verbose=False)

```

Gambar 4.176 Hasil Soal 4.

5. Soal 5

```

1 d_train_att
2
3 d_train_label=d_train[ 'CLASS' ]
4 d_test_label=d_test[ 'CLASS' ]

```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut :

```

In [8]: from sklearn import tree
... clftree = tree.DecisionTreeClassifier()
... clftree.fit(d_train_att, d_train_label)
Out[8]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated', random_state=None, splitter='best')

```

Gambar 4.177 Hasil Soal 5.

6. Soal 6

```

1 from sklearn import svm
2 clfsvm = svm.SVR(gamma = 'auto')
3 clfsvm.fit(d_train_att , d_train_label)
4
5 #%%soal 5
6
7 from sklearn import tree
8 clftree = tree.DecisionTreeClassifier()
9 clftree.fit(d_train_att , d_train_label)
10
11 #%%soal 6/
12
13 from sklearn.metrics import confusion_matrix
14 pred_labels=clftree.predict(d_test)
15 cm=confusion_matrix(d_test_label , pred_labels)
16
17 #%%
18
19 import matplotlib.pyplot as plt
20
21 def plot_confusion_matrix(cm,
22

```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```

In [11]: import matplotlib.pyplot as plt
...
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:     cm

```

Gambar 4.178 Hasil Soal 6.

7. Soal 7

```

1 title='Confusion matrix' ,
2 cmap=plt.cm.Blues):
3
4 if normalize:
5     cm = cm.astype('float') / cm.sum( axis=1)[:, np.
newaxis]
6     print("Normalized confusion matrix")
7 else:
8     print('Confusion matrix, without normalization')

```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan

menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```
In [12]: from sklearn.model_selection import cross_val_score
...
...
scores=cross_val_score(clftree,d_train_att,d_train_label,cv=5)
...
...
skor_rata2=scores.mean()
...
...
skoresd=scores.std()
```

Gambar 4.179 Hasil Soal 7.

8. Soal 8

```
1 print(cm)
2
3 #%%soal 7
4
5 from sklearn.model_selection import cross_val_score
6
7 scores=cross_val_score(clftree ,d_train_att ,d_train_label ,cv
8 =5)
9
10 skor_rata2=scores .mean()
11 skoresd=scores .std()
12
13 #%%soal 8 /
14 max_features_opts = range(5, 50, 5)
15 #membuat max_features_opts sebagai variabel untuk membuat
16 #membuat n_estimators_opts sebagai variabel untuk membuat
17 #membuat n_estimators_opts sebagai variabel untuk membuat
range 10,200,20
```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```
Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 100, accuracy: 0.33 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 38, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max features: 38, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 38, num estimators: 98, accuracy: 0.45 (+/- 0.04)
Max features: 38, num estimators: 100, accuracy: 0.45 (+/- 0.03)
Max features: 38, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 38, num estimators: 180, accuracy: 0.45 (+/- 0.03)
Max features: 38, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 38, num estimators: 199, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 100, accuracy: 0.34 (+/- 0.03)
Max features: 35, num estimators: 50, accuracy: 0.43 (+/- 0.02)
Max features: 35, num estimators: 98, accuracy: 0.43 (+/- 0.02)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.02)
Max features: 35, num estimators: 150, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 199, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 180, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 199, accuracy: 0.46 (+/- 0.04)
Max features: 40, num estimators: 100, accuracy: 0.34 (+/- 0.02)
Max features: 40, num estimators: 50, accuracy: 0.43 (+/- 0.02)
Max features: 40, num estimators: 98, accuracy: 0.43 (+/- 0.02)
Max features: 40, num estimators: 100, accuracy: 0.43 (+/- 0.02)
Max features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.04)
Max features: 40, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max features: 40, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 40, num estimators: 199, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 180, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max features: 40, num estimators: 199, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 100, accuracy: 0.34 (+/- 0.02)
Max features: 45, num estimators: 50, accuracy: 0.43 (+/- 0.02)
Max features: 45, num estimators: 98, accuracy: 0.43 (+/- 0.02)
Max features: 45, num estimators: 100, accuracy: 0.43 (+/- 0.02)
Max features: 45, num estimators: 150, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 199, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 180, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max features: 45, num estimators: 199, accuracy: 0.46 (+/- 0.04)
```

Gambar 4.180 Hasil Soal 8.

4.12.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundException: [Error: 2] File b'F://Semester 6/Artificial Intelligence/Tugas 4/src/fenny.csv' does not exist. b'F://Semester 6/Artificial Intelligence/Tugas 4/src/fenny.csv'
```

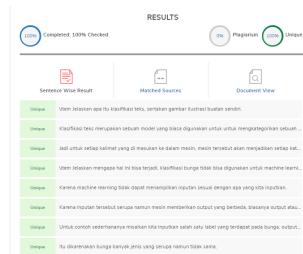
Gambar 4.181 SyntaxError

2. Cara Penangan Error

- **SyntaxError**

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.12.4 Bukti Tidak Plagiat



Gambar 4.182 Bukti Tidak Melakukan Plagiat Chapter 4

4.12.5 Link Youtube

<https://youtu.be/WA8JwD2lzPw>

BAB 5

CHAPTER 5

5.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 5.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

5.1.1 Teori

5.1.2 Praktek

5.1.3 Penanganan Error

5.1.4 Bukti Tidak Plagiat



Gambar 5.2 Kecerdasan Buatan.

BAB 6

CHAPTER 6

6.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 6.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

6.1.1 Teori

6.1.2 Praktek

6.1.3 Penanganan Error

6.1.4 Bukti Tidak Plagiat



Gambar 6.2 Kecerdasan Buatan.

BAB 7

CHAPTER 7

7.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 7.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

7.1.1 Teori

7.1.2 Praktek

7.1.3 Penanganan Error

7.1.4 Bukti Tidak Plagiat



Gambar 7.2 Kecerdasan Buatan.