

BAB 1

CHAPTER 1

1.1 Kaka Kamaludin

1.1.1 Teori

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

- Definisi Kecerdasan Buatan

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). AI sendiri merupakan suatu cabang dalam bisnis sains komputer sains dimana mengkaji tentang bagaimana cara untuk melengkapi sebuah komputer dengan kemampuan atau kepintaran layaknya atau mirip dengan yang dimiliki manusia. Sebagai contoh, sebagaimana komputer dapat berkomunikasi dengan pengguna baik menggunakan kata, suara maupun lain sebagainya. Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuiinya kemudian itulah yang disebut dengan kecerdasan buatan.

Kecerdasan Buatan adalah salah satu bidang studi yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara lebih manusiawi dan lebih bisa di pahami oleh manusia. Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan testing dan perkembangan target analisa.

- Sejarah Kecerdasan Buatan

Pada tahun 1956, McCarthy yang sama mendirikan Konferensi Dartmouth di Hanover, New Hampshire. Peneliti terkemuka dalam teori kompleksitas, simulasi bahasa, hubungan antara keacakan dan pemikiran kreatif, jaringan saraf diundang. Tujuan dari bidang penelitian yang baru dibuat adalah untuk mengembangkan mesin yang dapat mensimulasikan setiap aspek kecerdasan. Itulah sebabnya Konferensi Dartmouth 1956 dianggap sebagai kelahiran Kecerdasan Buatan. Sejak saat itu, Kecerdasan Buatan telah hidup melalui decade kemuliaan dan cemoohan, yang dikenal luas sebagai musim panas dan musim dingin AI. Musim panasnya ditandai dengan optimism dan dana besar, sedangkan musim dinginnya dihadapkan dengan pemotongan dana, ketidakpercayaan dan pesimisme.

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stanford.

- Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa

AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survei tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

AI Summer 1 (1956-1973) KOnferensi Dartmounth diikuti oleh 17 tahun kemajuan luar biasa. Proyek penelitian yang dilakukan di MIT, universitas di Edinburgh, Stanford dan Carnegie Mellon menerima dana besar-besaran, yang akhirnya membawa hasil. Selama tahun-tahun itulah komputer pemrograman mulai melakukan masalah aljabar, membuktikan teorema geometris, memahami dan menggunakan sintaks dan tata bahasa Inggris. Terlepas dari ditinggalkannya koneksiisme dan terjemahan mesin yang gagal, yang menunda penelitian Natural Language Processing (NLP) selama bertahun-tahun, banyak prestasi dari masa lalu yang membuat sejarah. Berikut ini beberapa diantaranya : Pelopor pembelajaran mesin, Ray Solomonoff meletakkan dasar-dasar teori metematika AI, memperkenalkan metode Bayesian universal untuk inferensi dan prediksi induktif Thomas Evans menciptakan program ANALOGI heuristik, yang memungkinkan komputer memecahkan masalah geometri analogi Unimation, perusahaan robotika pertama didunia, menciptakan robot industri Unimate, yang bekerja pada jalur perakitan modil Genenral Motors. Joseph Weizenbaum membangun ELIZA-program interaktif yang dapat membawa percakapan dalam bahasan Inggris tentang topik apapun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendikia-program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan Computeks and Thought, kumpulan artikel pertama tentang AI.

2. Definisi Supervised Learning, Klasifikasi, Regresi, Unsupervised Learning, Data Set, Training Set dan Testing Set
 - Definisi Supervised Learning

Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal". Supervised Learning menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk mendukung penilaian dimasa depan. Chatbots, mobil self-driving, program pengenalan wajah, sistem pakar dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Supervised Learning sebagian besar terkait dengan AI berbasis pengambilan tetapi mereka juga mungkin mampu menggunakan model pembelajaran generatif. Data pelatihan untuk pembelajaran yang diawasi mencakup serangkaian contoh dengan subjek input berpasangan dan output yang diinginkan (yang juga disebut sebagai sinyal pengawasan).

- Klasifikasi

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah. Label (y) umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran mesin dan statistik, klasifikasi adalah pendekatan pembelajaran yang diawasi di mana program komputer belajar dari input data yang diberikan kepadanya dan kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pengamatan baru. Kumpulan data ini mungkin hanya bersifat dua kelas (seperti mengidentifikasi apakah orang tersebut berjenis kelamin laki-laki atau perempuan atau bahwa surat itu spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, identifikasi metrik, klasifikasi dokumen dll.

- Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berke-

lanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

- Unsupervised Learning

Unsupervised Learning adalah pelatihan algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan. Dalam Unsupervised Learning, sistem AI dapat mengelompokkan informasi yang tidak disortir berdasarkan persamaan dan perbedaan meskipun tidak ada kategori yang disediakan.

- Data set

Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory. Strukturnya mirip dengan data di database, namun bedanya dataset berisi koleksi dari data table dan data relation mendapatkan data yang tepat berarti mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda prediksi; yaitu data yang berisi sinyal tentang peristiwa yang Anda pedulikan. Data harus diselaraskan dengan masalah yang Anda coba selesaikan. Gambar kucing tidak terlalu berguna ketika Anda sedang membangun sistem identifikasi wajah. Memverifikasi bahwa data selaras dengan masalah yang ingin Anda selesaikan harus dilakukan oleh ilmuwan data. Jika Anda tidak memiliki data yang tepat, maka upaya Anda untuk membangun solusi AI harus kembali ke tahap pengumpulan data.

- Training Set

Training Set adalah set digunakan oleh algoritma klassifikasi . Dapat dicontohkan dengan : decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier. Menjalankan pelatihan yang diatur melalui jaringan saraf mengajarkan pada net cara menimbang berbagai fitur, menyesuaikan koefisien berdasarkan kemungkinan mereka meminimalkan kesalahan dalam hasil Anda. Koefisien-koefisien tersebut, juga dikenal sebagai parameter, akan terkandung dalam tensor dan bersama-sama mereka disebut model, karena mereka mengkodekan model data yang mereka latih. Mereka adalah takeaways paling penting yang akan Anda dapatkan dari pelatihan jaringan saraf.

- Testing set

Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Ini berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakan sam-pai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda terhadap pengambilan sampel acak akhir ini. Hasil yang dihasilkannya harus memvalidasi bahwa jaring Anda secara akurat mengenali gambar, atau mengenalinya seti-daknya [x] dari jumlah tersebut. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan, lihat hyperparameter yang Anda gunakan untuk menyetel jaringan, serta kualitas data Anda dan lihat teknik pra-pemrosesan Anda.

1.1.2 Praktek

1. Instalasi library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer
 - Buka prompt anaconda lalu ketikkan conda scikit-learn
 - Biarkan proses berjalan hingga selesai

```
(base) C:\Users\root>conda install scikit-learn
```

Gambar 1.1 Install library scikit

```
(base) C:\Users\root>conda install scikit-learn
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

environment location: C:\Users\root\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be downloaded:
  package          |      build
  -----          | -----
  conda-4.8.2     |    py37_0      2.8 MB
  -----
  Total:          2.8 MB

The following packages will be UPDATED:
  conda           4.7.12-py37_0 --> 4.8.2-py37_0

Proceed ([y]/n)? y

Downloading and Extracting Packages
conda-4.8.2       | 2.8 MB  | #####| ################################# | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Gambar 1.2 Proses jika berhasil

2. Mencoba Loading an example dataset, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

- Ketikkan script berikut di spyder, lalu run

```

1 #Import fungsi datasets dari library sklearn
2 from sklearn import datasets
3
4 #Memasukkan data dari datasets iris ke variable iris
5 iris = datasets.load_iris()
6
7 #Memasukkan data dari datasets digits ke variable digits
8 digits = datasets.load_digits()
9
10 #Menampilkan data dari datasets digits ke console
11 print(digits.data)

```

```

In [1]: runfile('E:/MyLatex/src/1174067/src1/test.py', wdir='E:/MyLatex/src/1174067/src1')
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0.  10. ... 12.  1.  0.]]

```

Gambar 1.3 hasil test.py

3. Mencoba Learning and predicting, menjelaskan maksud dari tulisan tersebut dan mengartikan perbaris

```

1
2 #Import fungsi datasets dari library sklearn
3 from sklearn import datasets
4
5 #Memasukkan data dari datasets iris ke variable iris
6 iris = datasets.load_iris()
7
8 #Memasukkan data dari datasets digits ke variable digits
9 digits = datasets.load_digits()
10
11 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
12     algoritma classification yang akan diambil dari Scikit –
13     Learn .
14 from sklearn import svm
15
16 #Mendeklarasikan suatu value yang bernama clf yang berisi
17     gamma.
18 clf = svm.SVC(gamma=0.001, C=100.)
19
20 #Estimator clf (for classifier)
21 clf.fit(digits.data[:-1], digits.target[:-1])

```

```

19
20 #Menunjukkan prediksi angka baru
21 hasil = clf.predict(digits.data[-1:])
22
23 #Menampilkan
24 print(hasil)

```

Name	Type	Size	Value
digits	utils.Bunch	5	Bunch object of sklearn.utils module
hasil	int32	(1,)	[8]
iris	utils.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.4 variabel explorer test2.py

4. Mencoba Model persistence, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

```

1 #%%Cara Dump Pertama
2
3
4
5 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
   algoritma classification yang akan diambil dari Scikit-
   Learn.
6 from sklearn import svm
7 #Import fungsi datasets dari library sklearn
8 from sklearn import datasets
9
10 #Mendefinisikan clf dengan fungsi svc dari library svm
11 clf = svm.SVC()
12
13 #Mengisi variable x dan y dengan data dari datasets
14 X, y = datasets.load_iris(return_X_y=True)
15
16 #Estimator clf (for classifier)
17 clf.fit(X, y)
18
19 #Mengimport Library pickle
20 import pickle
21
22 #Menyimpan hasil dari clf kedalam sebuah dump
23 s = pickle.dumps(clf)
24
25 #Memanggil dump yang dihasilkan pickle lalu memasukkan hasil
   dumpnya ke variable
26 clf2 = pickle.loads(s)
27
28
29 #Memprediksi angka yang akan muncul
30 clf2.predict(X[0:1])
31

```

```

32 #Menampilkan data prediksi
33 print(y[0])
34
35 ##%Cara Dump Kedua
36 # Digunakan untuk memanggil class svm dari library sklearn
37 from sklearn import svm
38
39 # Digunakan untuk class datasets dari library sklearn
40 from sklearn import datasets
41
42 # membuat variabel clf , dan memanggil class svm dan fungsi
SVC
43 clf = svm.SVC()
44
45 #Mengambil dataset iris dan mengembalikan nilainya.
46 X, y = datasets.load_iris(return_X_y=True)
47
48 #Perhitungan nilai label
49 clf.fit(X, y)
50
51
52 #memanggil class dump dan load pada library joblib
53 from joblib import dump, load
54
55
56 #Menyimpan model kedalam 1174067.joblib
57 dump(clf, '1174067.joblib')
58
59 #Memanggil model 1174067
60 hasil = load('1174067.joblib')
61 hasil.predict(X[0:1])
62
63 # Menampilkan Model yang dipanggil sebelumnya

```

Name	Type	Size	Value
X	float64	(150, 4)	[[5.1 3.5 1.4 0.2] [4.9 3. 1.4 0.2]]
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module
y	int32	(150,)	[0 0 0 ... 2 2 2]

Gambar 1.5 variabel explorer test3.py

5. Mencoba Conventions, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

```

1
2 # memanggil library numpy dan dibuat alias np
3 import numpy as np
4
5 #Memanggil class random_projection pada library sklearn

```

```

6 from sklearn import random_projection
7
8 #Membuat variabel rng , dan mendefinisikan np , fungsi random dan
9     attr RandomState kedalam variabel
10    rng = np.random.RandomState(0)
11
12 # membuat variabel X, dan menentukan nilai random dari 10 –
13     2000
13 X = rng.rand(10, 2000)
14
14 #menyimpan hasil nilai random sebelumnya , kedalam array , dan
15     menentukan typedatanya sebagai float32
15 X = np.array(X, dtype='float32')
16
16 # Mengubah data tipe menjadi float64
17 X.dtype
17
18
19
20 #membuat variabel transformer , dan mendefinisikan
21     classrandom_projection dan memanggil fungsi
21     GaussianRandomProjection
21 transformer = random_projection.GaussianRandomProjection()
22
22 # membuat variabel baru dan melakukan perhitungan label pada
23     variabel X
23 X_new = transformer.fit_transform(X)
24
24 # Mengubah data tipe menjadi float64
25 X_new.dtype
26
26 # Menampilkan isi variabel X_new
27 print(X_new)
27

```

Name	Type	Size	Value
X	float32	(10, 2000)	[10.5488135 0.71518934 0.60276335 ... 0.4801078 0.64386404 0.5017731 ...]
X_new	float64	(10, 1973)	[-0.74506698 0.45457383 0.88209808 ... -0.00255147 -0.19283521 -0 ...]
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module
y	int32	(150,)	[0 0 0 ... 2 2 2]

Gambar 1.6 variabel explorer test4.py

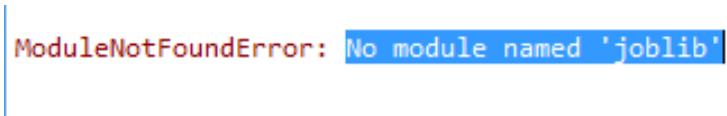
```
In [5]: runfile('E:/MyLatex/src/1174067/src1/test4.py', wdir='E:/MyLatex/src/1174067/src1')
[[ -0.74506698  0.45457383  0.88209808 ... -0.00255147 -0.19283521
-0.0999184 ]
[-0.74474565 -0.07596842  0.75740284 ...  0.92772654 -0.08133772
-0.12007804]
[-0.65903242  0.32393276  1.35294319 ...  0.69425209 -0.01671971
0.49786433]
...
[-0.44103442  0.42938771  0.53121863 ...  0.35585446  0.13729904
0.23401575]
[-0.70748146 -0.29337808  1.12836231 ...  0.81903801 -0.35044182
0.03876353]
[-0.53525128  0.6922521   0.615109    ...  0.75372493  0.10521824
0.2622668 ]]

In [6]: |
```

Gambar 1.7 hasil test4.py

1.1.3 Penanganan Error

1. Screenshoots Error



```
ModuleNotFoundError: No module named 'joblib'
```

Gambar 1.8 error

2. Jenis Error

- Not Module name "joblib"

3. Cara Penanganan Error

- install library joblib

1.1.4 Bukti tidak plagiat

1.1.5 Link Youtube

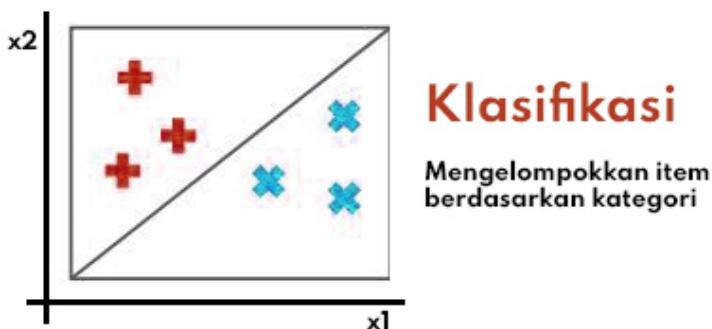
<https://youtu.be/FDQopV2LUIo>

BAB 2

CHAPTER 2

2.1 Kaka Kamaludin(1174067)

2.1.1 Teori



Gambar 2.1 Binary Classification

2.1.1.1 Binary Classification Binary Classification (Klasifikasi Biner) adalah sebuah tugas yang mengklasifikasikan elemen-elemen dari himpunan yang diberikan ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi. Konteks yang membutuhkan keputusan apakah suatu item memiliki sifat kualitatif atau tidak, beberapa karakteristik tertentu, atau beberapa klasifikasi biner khas meliputi:

1. Tes medis

untuk menentukan apakah pasien memiliki penyakit tertentu atau tidak - properti klasifikasi adalah keberadaan penyakit.

2. Metode uji "lulus atau gagal"

Kontrol kualitas di pabrik, yaitu memutuskan apakah suatu spesifikasi telah atau belum terpenuhi - klasifikasi Go/no go.

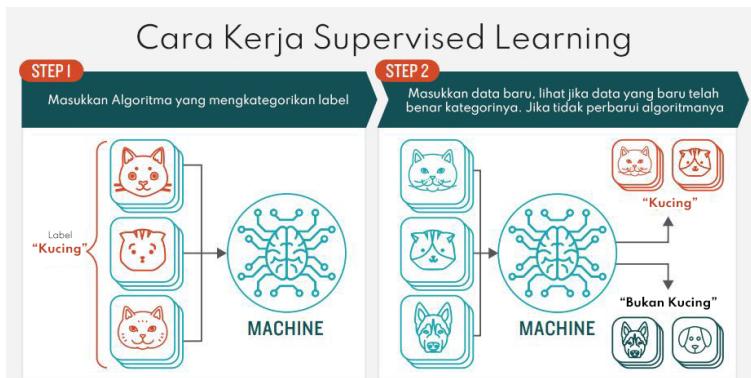
3. Pengambilan informasi

memutuskan apakah suatu halaman atau artikel harus ada dalam hasil pencarian atau tidak - properti klasifikasi adalah relevansi artikel.

2.1.1.2 Supervised Learning , Unsupervised Learning dan Clustering

1. Supervised Learning

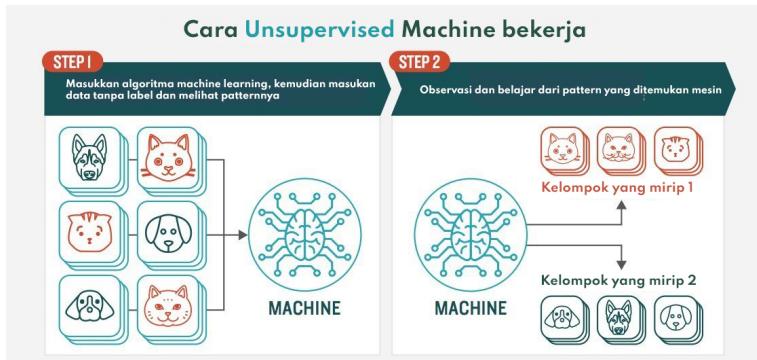
Supervised learning adalah suatu pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksud disini ada supervisornya adalah label di tiap data nya. Label maksudnya adalah tag dari data yang ditambahkan dalam machine learning model. Contohnya gambar kucing di tag "kucing" di tiap masing masing image kucing dan gambar anjing di tag "anjing" di tiap masing gambar anjing.



Gambar 2.2 Supervised Learning

2. Unsupervised Learning

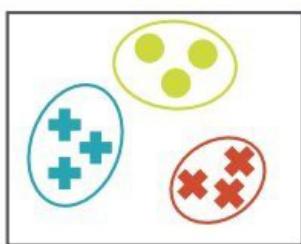
Unsupervised learning menggunakan kemiripan dari attribut yang dimiliki suatu item. Jika attribut dan sifat dari data yang diekstrak memiliki kemiripan, maka akan dikelompokkan (clustering). Sehingga hal ini akan menimbulkan kelompok (cluster). Jumlah cluster bisa unlimited. Dari kelompok-kelompok itu model dilabelkan, dan jika data baru mau diprediksi, maka akan dicocokkan dengan data kelompok yang mirip featurenya.



Gambar 2.3 Unsupervised Learning

3. Clustering

Clustering adalah sebuah metode untuk membedakan data-data menjadi sebuah kumpulan dari group yang isinya merupakan data yang mirip setiap grupnya. Basisnya dapat berupa kesamaan atau perbedaan dari setiap grup tersebut.



CLUSTERING

Melihat kemiripan dari setiap item dan mengelompokkannya ke sebuah group

Gambar 2.4 Clustering

2.1.1.3 Evaluasi dan Akurasi Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur tingkat akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan

dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan (confusion matrix). Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

	Hasil Prediksi "Apel"	Hasil Prediksi "Jeruk"
True "Apel"	20	5
True "Jeruk"	3	22

Gambar 2.5 Evaluasi

2.1.1.4 Cara Membuat Confusion Matrix Confusion Matrix merupakan metode untuk menghitung akurasi pada data mining atau Sistem Pendukung Keputusan. Untuk menggunakan Confusion Matrix, ada 4 istilah sebagai hasil proses dari klasifikasi. Diantaranya adalah:

- True Positive: Data positif yang terdeteksi memiliki hasil benar
- False Positive: Data Positif yang terdeteksi memiliki hasil salah
- True Negative: Data negatif yang terdeteksi memiliki hasil benar
- False Negative: Data negatif yang terdeteksi memiliki hasil salah

	Data Asli 1 (positif)	Data Asli 0 (negatif)
Prediksi 1 (positif)	True Positive (TP)	False Positive (FP)
Prediksi 0 (negatif)	False Negative (FN)	True Negative (TN)

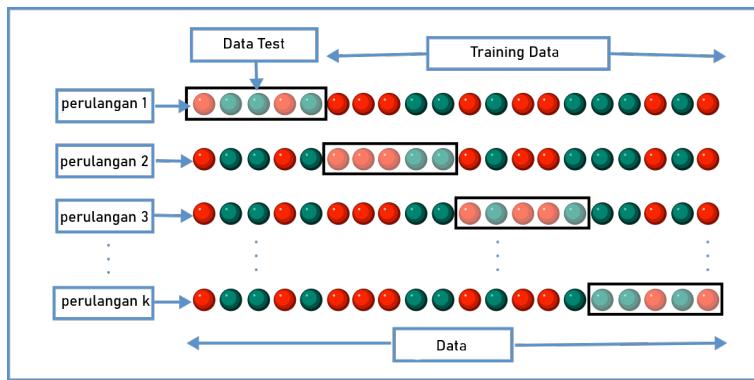
Gambar 2.6 Contoh Confusion Matrix

2.1.1.5 Bagaimana K-fold cross validation bekerja

1. Total instance dibagi menjadi N bagian.
2. Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
3. Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
4. Fold yang kedua adalah bagian kedua, yang menjadi data uji (testing data) dan sisanya training data.
5. Kemudian hitung akurasi berdasarkan porsi data tersebut.

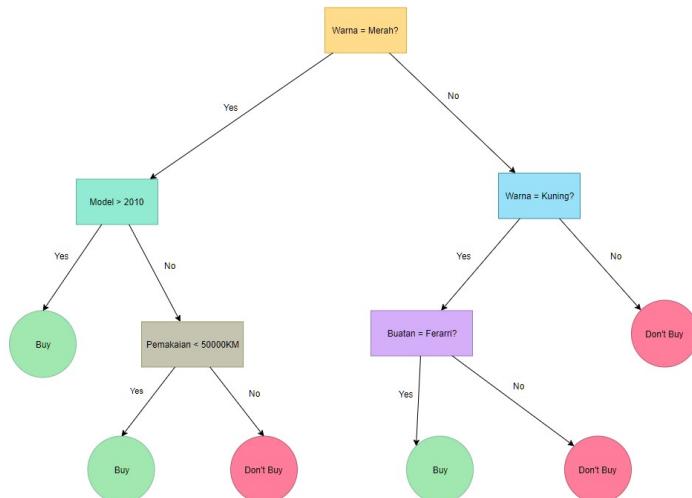
6. Dan seterusnya hingga habis mencapai fold ke-K.

7. Terakhir hitung rata-rata akurasi K buah.



Gambar 2.7 Contoh K-Fold Cross Validation

2.1.1.6 Apa itu Decision Tree Decision Tree adalah sebuah struktur yang menentukan keputusan dan setiap konsekuensinya. Hasil dari setiap struktur biasanya menggunakan jawaban (True dan False) atau cabang lain yang akan menjadi pohon selanjutnya. Setiap keputusan diantaranya akan membandingkan kondisi yang diberikan kepada struktur untuk dibandingkan kondisi apa saja yang sudah didapat pada sistem tersebut.



Gambar 2.8 Contoh Decision Tree membeli mobil

2.1.1.7 Information Gain dan Entropi

- Information Gain

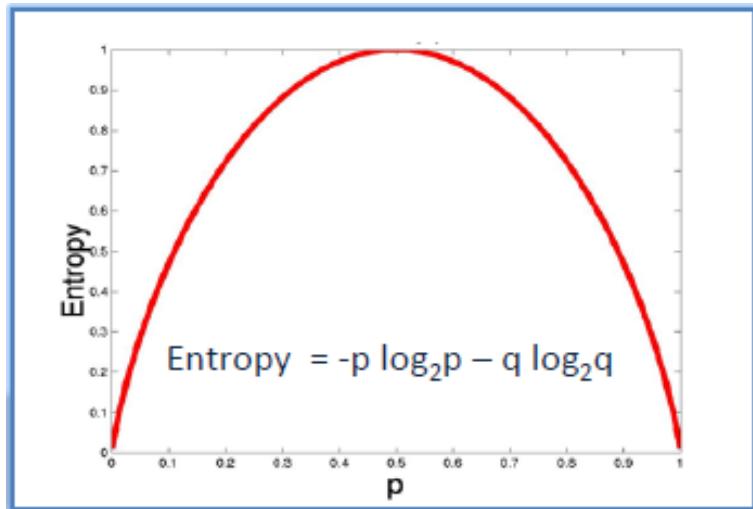
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.9 Information Gain

- Entropi

Entropi merupakan pengukuran sebuah data dan validnya data tersebut untuk dapat digunakan sebagai informasi yang akan dimasukkan ke Information Gain. Entropi menilai sebuah obyek berdasarkan kebutuhan di dunia nyata dan pengaruh pada sistem yang akan digunakan.



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

Gambar 2.10 Entropi

2.1.2 Praktek

Tugas anda adalah, dataset ganti menggunakan student-mat.csv dan mengganti semua nama variabel dari kode di bawah ini dengan nama-nama makanan (NPM mod 3=0), kota (NPM mod 3=1), buah (NPM mod 3=2)

```
1
2 # In [2]
```

2.1.2.1 Nomor 1

```
1 tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1) #Menghilangkan data
      G1 G2 dan G3
2 tokyo.head() #Menampilkan data
```

The screenshot shows a Jupyter Notebook interface. On the left, there is a code cell containing the following Python code:

```
# coding: utf-8
#
# Created on Sun Mar 8 12:50:42 2020
# @author: RIN
# @file: student-mat.py
# @Software: PyCharm
# Description: 2 kelas ubah nongresoren nomen data
# 21210051 X S. Mulyana
# 18/3/2020
# Import pandas as pd
# Import library
# df = pd.read_csv('dataset/student-mat.csv', sep=';')
# df.head()
```

The output of this cell is shown in the right pane, which includes a 'Help' box and a 'Variable explorer' window.

Gambar 2.11 Nomor 1

2.1.2.2 Nomor 2

```

1 # use one-hot encoding on categorical columns
2 tokyo = pd.get_dummies(tokyo, columns=['sex', 'school', 'address',
3     , 'famsize', 'Pstatus', 'Mjob', 'Fjob',
4     , 'reason', 'guardian', 'schoolsupt',
5     , 'famsup', 'paid', 'activities',
6     , 'nursery', 'higher', 'internet', ,
7     'romantic'])
8 tokyo.head()

```

The screenshot shows the Jupyter Notebook interface with two code cells. The first cell contains the code for creating one-hot encoding. The second cell shows the output of `tokyo.head()`, displaying the first 5 rows of the dataset. The output includes columns like 'sex', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsupt', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', and 'romantic'. The data values are represented by integers (0 or 1) corresponding to the categories.

Gambar 2.12 Nomor 2

2.1.2.3 Nomor 3

```

1 # In [4]:
2 # shuffle rows
3 tokyo = tokyo.sample(frac=1) #Mengambil data sample dari tokyo
4 # split training and testing data
5 tokyo_train = tokyo[:500] #Membagi data untuk training
6 tokyo_test = tokyo[500:] #Membagi data untuk test

```

The screenshot shows the Jupyter Notebook interface with three code cells. The first cell contains the code for shuffling the rows. The second cell contains the code for splitting the data into training and testing sets. The third cell shows the output of `tokyo.head()`, which is identical to the output in Gambar 2.12, showing the first 5 rows of the dataset with one-hot encoding applied.

Gambar 2.13 Nomor 3

2.1.2.4 Nomor 4

```

1 tokyo_train_att = tokyo_train.drop(['pass'], axis=1) #Meghapus
2     data yang telah pass dan memasukkannya
3 tokyo_train_pass = tokyo_train['pass'] #Mengambil data yang pass
4     saja
5 tokyo_test_att = tokyo_test.drop(['pass'], axis=1) #Meghapus data
6     yang telah pass dan memasukkannya
7 tokyo_test_pass = tokyo_test['pass'] #Mengambil data yang pass
8     saja

```

```

6
7 tokyo_att = tokyo.drop(['pass'], axis=1) #Menghapus data yang
     telah pass dan memasukkannya
8 tokyo_pass = tokyo['pass'] #Mengambil data yang pass saja
9
10 # number of passing students in whole dataset:
11 import numpy as np #Mengimport library numpy sebagai np
12 print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(
     tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)
    )) #Menampilkan data
13
14 # In [5]:
15 # fit a decision tree
16 from sklearn import tree #import Decision tree dari library
     sklearn
17 kyoto = tree.DecisionTreeClassifier(criterion="entropy",
     max_depth=5) #Membuat decision tree dengan maximal depthnya 5
18 kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass) #Memasukkan
     data yang akan dijadikan decision treenya

```

```

# In [21]:
tokyo = tokyo.sample(frac=0.5)
tokyo_train = tokyo[0:100]
tokyo_test = tokyo[100:150]

tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
tokyo_train_pass = tokyo_train['pass']

tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
tokyo_test_pass = tokyo_test['pass']

tokyo_att = tokyo_att.sample(frac=0.5)
tokyo_pass = tokyo_pass.sample(frac=0.5)

tokyo_att = tokyo_att.reset_index(drop=True)
tokyo_pass = tokyo_pass.reset_index(drop=True)

# number of passing students in whole dataset:
print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))

```

```

ModuleNotFoundError: No module named 'graphviz'
In [22]:
In [22]: tokyo = tokyo.sample(frac=0.5)
tokyo_train = tokyo[0:100]
tokyo_test = tokyo[100:150]
tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
tokyo_train_pass = tokyo_train['pass']
tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
tokyo_test_pass = tokyo_test['pass']

tokyo_att = tokyo_att.reset_index(drop=True)
tokyo_pass = tokyo_pass.reset_index(drop=True)

tokyo_att = tokyo_att.sample(frac=0.5)
tokyo_pass = tokyo_pass.sample(frac=0.5)

tokyo_att = tokyo_att.reset_index(drop=True)
tokyo_pass = tokyo_pass.reset_index(drop=True)

# number of passing students in whole dataset:
print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))

```

```

Passing: 164 out of 200 (82.00%)

```

Gambar 2.14 Nomor 4

2.1.2.5 Nomor 5

```

1 # In [6]:
2 # visualize tree
3 import graphviz #Mengimport Library Graphviz untuk
     memvisualisasikan decision tree
4 dot_data = tree.export_graphviz(kyoto, out_file=None, label="all",
     , impurity=False, proportion=True,
     feature_names=list(
      tokyo_train_att), class_names=["fail", "pass"],

```

In [24]: `from sklearn import tree`
`...: kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)`
`...: kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass)`

Gambar 2.15 Nomor 5

2.1.2.6 Nomor 6

```

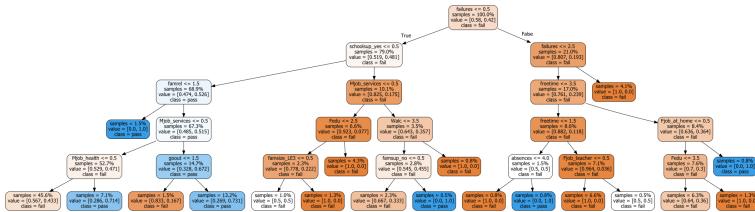
1 graph #Menampilkan graph menggunakan graphviz
2
3

```

```

4 # In [7]:
5 # save tree
6 tree.export_graphviz(kyoto, out_file="student-performance.dot",
7     label="all", impurity=False, proportion=True,
8         feature_names=list(tokyo_train_att),
9         class_names=["fail", "pass"],
10            filled=True, rounded=True) #Digunakan untuk
mengexport graph tree tadi yang telah kita buat

```



Gambar 2.16 Nomor 6

2.1.2.7 Nomor 7

```

1 # In [8]:
2 kyoto.score(tokyo_test_att, tokyo_test_pass) #Menghitung prediksi
    nilai yang akan datang dimasa depan
3
4
5 # In [9]:

```

```
In [32]: tree.export_graphviz(kyoto, out_file="student-performance.dot", label="all", impurity=False, proportion=True,
...:     feature_names=list(tokyo_train_att), class_names=["fail", "pass"],
...:     filled=True, rounded=True) #digunakan untuk mengexport graph tree tadi yang telah kita buat
```

Gambar 2.17 Nomor 7

2.1.2.8 Nomor 8

```

1 # show average score and +/- two standard deviations away (
2     covering 95% of scores)
3 print("Accuracy: %0.2f (+/- %0.2f)" % (nagoya.mean(), nagoya.std()
4     () * 2)) #Menampilkan data nilai dan +/- dari dua standar
    deviasi

```

In [60]: kyoto.score(tokyo_test_att, tokyo_test_pass)
Out[60]: 0.6778523489932886

Gambar 2.18 Nomor 8

2.1.2.9 Nomor 9

```

1 # In [10]:
2 for max_depth in range(1, 20): #Pengulangan menunjukkan seberapa
3     dalam tree itu
4     kyoto = tree.DecisionTreeClassifier(criterion="entropy",
5     max_depth=max_depth) #Membuat decision Tree
6     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
7     #Mendefinisikan nagoya yang isinya pembagian data menjadi 5
8     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
9         max_depth, nagoya.mean(), nagoya.std() * 2)) #Menampilkan
10    data nilai dan +/- dari dua standar deviasi

```

```

In [69]: from sklearn.model_selection import cross_val_score
...: nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (nagoya.mean(), nagoya.std() * 2))
Accuracy: 0.55 (+/- 0.10)

```

Gambar 2.19 Nomor 9

2.1.2.10 Nomor 10

```

1 # In [11]:
2 depth_acc = np.empty((19,3), float) #Membuat array baru
3 i = 0 #Membuat variable berisikan 0
4 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
5     kyoto = tree.DecisionTreeClassifier(criterion="entropy",
6     max_depth=max_depth)#Membuat decision Tree

```

```

In [70]: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, nagoya.mean(), nagoya.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.08)
Max depth: 3, Accuracy: 0.58 (+/- 0.12)
Max depth: 4, Accuracy: 0.57 (+/- 0.08)
Max depth: 5, Accuracy: 0.58 (+/- 0.09)
Max depth: 6, Accuracy: 0.57 (+/- 0.10)
Max depth: 7, Accuracy: 0.57 (+/- 0.15)
Max depth: 8, Accuracy: 0.54 (+/- 0.10)
Max depth: 9, Accuracy: 0.54 (+/- 0.11)
Max depth: 10, Accuracy: 0.58 (+/- 0.11)
Max depth: 11, Accuracy: 0.56 (+/- 0.10)
Max depth: 12, Accuracy: 0.60 (+/- 0.08)
Max depth: 13, Accuracy: 0.59 (+/- 0.05)
Max depth: 14, Accuracy: 0.58 (+/- 0.05)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.58 (+/- 0.08)
Max depth: 17, Accuracy: 0.60 (+/- 0.11)
Max depth: 18, Accuracy: 0.58 (+/- 0.08)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)

```

Gambar 2.20 Nomor 10

2.1.2.11 Nomor 11

```

1 depth_acc[i,1] = nagoya.mean() #Memasukkan data rata-rata
2 dari nagoya ke array depth_acc
2 depth_acc[i,2] = nagoya.std() * 2 #Memasukkan data akar 2
3 dari nagoya ke array depth_acc

```

```

3     i += 1
4
5 depth_acc
6
7
8 # In[12]:
9 import matplotlib.pyplot as plt #Menimport fungsi pyplot dari
10    library matplotlib sebagai plt
11 fig, ax = plt.subplots() #Membuat plot baru
12 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
13      #Mengisikan data plot
14 plt.show() #Menampilkan plot

```

```

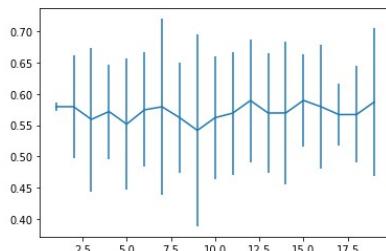
In [71]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = nagoya.mean()
...:     depth_acc[i,2] = nagoya.std() * 2
...:     i += 1
...:
...: depth_acc
Out[71]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.79654333e-01, 8.25005697e-02],
       [3.00000000e+00, 5.54241318e-01, 1.21808510e-01],
       [4.00000000e+00, 5.71964460e-01, 8.72318860e-02],
       [5.00000000e+00, 5.46678027e-01, 9.66616528e-02],
       [6.00000000e+00, 5.69561019e-01, 1.08113451e-01],
       [7.00000000e+00, 5.67030996e-01, 1.40406275e-01],
       [8.00000000e+00, 5.47030996e-01, 1.22447311e-01],
       [9.00000000e+00, 5.51966082e-01, 6.68884125e-02],
       [1.00000000e+01, 5.87314184e-01, 7.12478298e-02],
       [1.10000000e+01, 5.77124310e-01, 7.61119153e-02],
       [1.20000000e+01, 5.89719247e-01, 4.34452239e-02],
       [1.30000000e+01, 5.97569783e-01, 3.87484760e-02],
       [1.40000000e+01, 5.77026939e-01, 7.68881184e-02],
       [1.50000000e+01, 5.97347452e-01, 5.69404888e-02],
       [1.60000000e+01, 5.74720058e-01, 1.29008478e-01],
       [1.70000000e+01, 5.82282538e-01, 5.71762018e-02],
       [1.80000000e+01, 5.74720870e-01, 3.56163418e-02],
       [1.90000000e+01, 5.77250893e-01, 8.75345835e-02]])

```

Gambar 2.21 Nomor 11

2.1.2.12 Nomor 12

```
In [73]: import matplotlib.pyplot as plt #Menimport fungsi pyplot dari library matplotlib sebagai plt
      ...: fig, ax = plt.subplots() #Membuat plot baru
      ...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2]) #Mengisikan data plot
      ...: plt.show() #Menampilkan plot
```



Gambar 2.22 Nomor 12

2.1.3 Penanganan Error

2.1.3.1 *Error*

1. ModuleNotFoundError

```
Traceback (most recent call last):
  File "<ipython-input-25-af85b140ad99>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.23 ModuleNotFoundError

2.1.3.2 *Solusi*

1. Intall library Graphviz dengan cara download graphviz di google setelah instalasi buka anaconda prompt sebagai admin lalu mengetikkan
1 conda install graphviz

2.1.4 Link Youtube:

<https://www.youtube.com/playlist?list=PL4dhp4u89PHbhX9jrGyM3N12gmwhY3uIe>

BAB 3

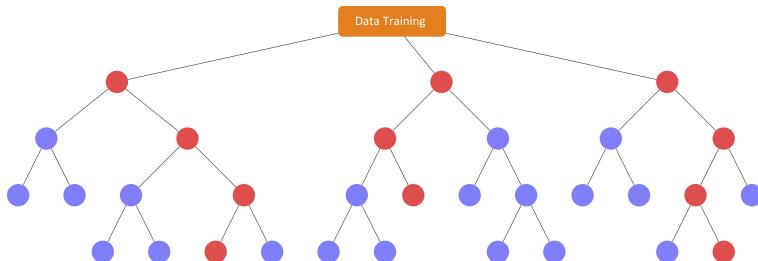
CHAPTER 3

3.1 Kaka Kamaludin (1174067)

3.1.1 Teori

3.1.1.1 *Apa Itu Random Forest*

Random Forest adalah sebuah algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakuakn training terhadap sempel data yang dimiliki. Semakin banyak dicision tree maka data yang di dapat akan semakin akurat.



Gambar 3.1 Random Forest

3.1.1.2 cara membaca dataset kasus dan artikan makna setiap file dan isi field masing-masing file.

- Pertama download dataset terlebih dahulu lalu buka dengan menggunakan software spyder guna melihat isi dari dataset tersebut.
- Data tersebut memiliki extensi file bernama .txt dan didalamnya terdapat class dari field.
- Misalnya saja pada data jenis burung memiliki file index dan angka, dimana index berisi angka yang memiliki makna berupa jenis burung.
- bahkan nama burung sedangkan field memiliki isi nilai berupa 0 dan 1 yang dimana sifatnya boolean atau Ya dan Tidak.
- Hal ini dikarenakan komputer hanya dapat membaca bilangan biner maka dari itu field yang di isikan berupa angka.
- Artinya angka 0 berarti tidak dan angka 1 berarti Ya.

3.1.1.3 Cross Validation

Cross Validation adalah sebuah teknik validasi model yang berfungsi untuk melakukan penilaian bagaimana hasil analisis statistik akan digeneralisasi ke data yang lebih independen. Cross validation digunakan dengan tujuan prediksi, dan bila kita ingin memperkirakan seberapa akurat model prediksi yang dilakukan dalam sebuah praktik. Tujuan dari cross validation yaitu untuk mendefinisikan dataset guna menguji dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

3.1.1.4 Arti score 44 % pada random forest, 27% pada decission tree dan 29 % dari SVM.

Dimana Score 44 % didapatkan dari hasil pengelahan dataset jenis burung. Dimana akan dilakukan proses pembagian data testing dan data training lalu

diproses dan menghasilkan score sebanyak 44 % dimana menjelaskan bahwa score tersebut digunakan sebagai pembanding dalam tingkat keakuratannya. Pada dicision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan dicision tree dibagi menjadi beberapa tree dan lalu disimpulkan untuk mendapatkan data yang akurat. Pada SVM akan memperoleh score sebanyak 29 % hal ini dikarenakan data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.

3.1.1.5 Cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Perhitungan Confusion Matriks dapat dilakukan dengan cara dibawah ini.

- Import librari Pandas, Matplotlib, dan Numpy.
- Buat variabel y actu yang berisikan data aktual.
- Buat variabel y pred berisikan data yang akan dijadikan sebagai prediksi.
- Buat variabel df confusion yang berisikan crosstab untuk membangun tabel tabulasi silang yang dapat menunjukkan frekuensi kemunculan kelompok data tertentu.
- Pada variabel df confusion definisikan lagi nama baris yaitu Actual dan kolomnya Predicted
- Kemudian definisikan suatu fungsi yang diberi nama plot confusion matrix yang berisikan pendefinisian confusion matrix dan juga akan diplotting.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 14 01:32:47 2020
4
5 @author: root
6 """
7 import numpy as np
8 import matplotlib.pyplot as plt
9 import pandas as pd
10 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
11                   ='Actual')
12 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
13                   ='Predicted')
14 df_confusion = pd.crosstab(y_actu, y_pred)
15 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual',
16                             ], colnames=[ 'Predicted'], margins=True)
17 def plot_confusion_matrix(df_confusion , title='Confusion
18                           matrix' , cmap=plt.cm.gray_r):
19     plt.matshow(df_confusion , cmap=cmap) # imshow
20     #plt.title(title)
21     plt.colorbar()
22     tick_marks = np.arange(len(df_confusion.columns))

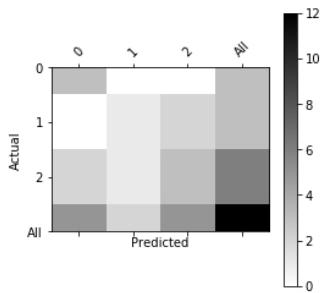
```

```

19     plt.xticks(tick_marks, df_confusion.columns, rotation=45)
20     plt.yticks(tick_marks, df_confusion.index)
21     #plt.tight_layout()
22     plt.ylabel(df_confusion.index.name)
23     plt.xlabel(df_confusion.columns.name)
24 plot_confusion_matrix(df_confusion)
25 plt.show()

```

In [1]: runfile('D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/KB3C/src/1174067/3/commatriks.py', wdir='D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/KB3C/src/1174067/3')



Gambar 3.2 Confusion Matriks

3.1.1.6 Apa itu voting pada random forest

Voting yaitu suara untuk setiap target yang diprediksi pada saat melakukan Random Forest dilakukan. Pertimbangkan target prediksi dengan voting tertinggi sebagai prediksi akhir dari algoritma random forest.

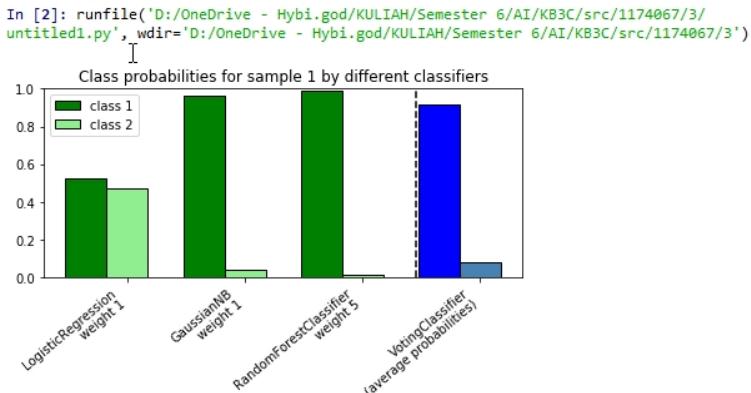
Untuk menggunakan Voting pada Random Forest dapat dilihat code ini:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 14 01:32:47 2020
4
5 @author: root
6 """
7
8 import numpy as np
9 import matplotlib.pyplot as plt
10 from sklearn.linear_model import LogisticRegression
11 from sklearn.naive_bayes import GaussianNB
12 from sklearn.ensemble import RandomForestClassifier
13 from sklearn.ensemble import VotingClassifier
14 clf1 = LogisticRegression(solver='lbfgs', max_iter=1000,
   random_state=123)
15 clf2 = RandomForestClassifier(n_estimators=100, random_state=123)
16 clf3 = GaussianNB()
17 X = np.array([[-1.0, -1.0], [-1.2, -1.4], [-3.4, -2.2], [1.1,
   1.2]])
18 y = np.array([1, 1, 2, 2])
19 eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2),
   ('gnb', clf3)], )

```

```
20         voting='soft',
21         weights=[1, 1, 5])
22 # predict class probabilities for all classifiers
23 probas = [c.fit(X, y).predict_proba(X) for c in (clf1, clf2, clf3
24 , eclf)]
25 # get class probabilities for the first sample in the dataset
26 class1_1 = [pr[0, 0] for pr in probas]
27 class2_1 = [pr[0, 1] for pr in probas]
28 # plotting
29 N = 4 # number of groups
30 ind = np.arange(N) # group positions
31 width = 0.35 # bar width
32 fig, ax = plt.subplots()
33 # bars for classifier 1-3
34 p1 = ax.bar(ind, np.hstack(([class1_1[:-1], [0]])), width,
35             color='green', edgecolor='k')
36 p2 = ax.bar(ind + width, np.hstack(([class2_1[:-1], [0]])), width
37             ,
38             color='lightgreen', edgecolor='k')
39 # bars for VotingClassifier
40 p3 = ax.bar(ind, [0, 0, 0, class1_1[-1]], width,
41             color='blue', edgecolor='k')
42 p4 = ax.bar(ind + width, [0, 0, 0, class2_1[-1]], width,
43             color='steelblue', edgecolor='k')
44 # plot annotations
45 plt.axvline(2.8, color='k', linestyle='dashed')
46 ax.set_xticks(ind + width)
47 ax.set_xticklabels(['LogisticRegression\nnweight 1',
48                     'GaussianNB\nnweight 1',
49                     'RandomForestClassifier\nnweight 5',
50                     'VotingClassifier\nn(average probabilities)'],
51                    rotation=40,
52                    ha='right')
53 plt.ylim([0, 1])
54 plt.title('Class probabilities for sample 1 by different
55 classifiers')
56 plt.legend([p1[0], p2[0]], ['class 1', 'class 2'], loc='upper
57 left')
58 plt.tight_layout()
59 plt.show()
```



Gambar 3.3 Voting Random Matriks

3.1.2 Praktek

3.1.2.1 Nomor 1

```

1 # In [0]
2 import pandas as KK # Melakukan import library pandas menjadi
   nama sendiri yaitu KK
3
4 buah = {"Nama buah" : ['apel', 'ceri', 'boom', 'melon']} # Membuat
   varibel yang bernama aplikasi, dan mengisi dataframanya
   dengan nama nama aplikasi koding
5 x = KK.DataFrame(buah) # Membuat variabel x yang akan membuat
   DataFrame dari library pandas yang akan memanggil variabel
   aplikasi.
6 print (' KK makan buah: ' + x) #print hasil dari x

```

In [7]: runfile('D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/KB3C/src/1174067/3/1174067.py', wdir='D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/KB3C/src/1174067/3')

I	Nama Aplikasi
0	KK pake aplikasi: VSCode
1	KK pake aplikasi: Atom
2	KK pake aplikasi: Sublime
3	KK pake aplikasi: Notepad++

Gambar 3.4 Membuat Aplikasi pakai pandas

3.1.2.2 Nomor 2

```

1 import numpy as KK #Melakukan import library numpy menjadi nama
   sendiri yaitu KK
2
3 matrix_x = KK.eye(10) #Membuat sebuah matrix pake numpy dengan
   menggunakan fungsi eye

```

```

4 matrix_x #Mendeklarasikan matrix_x yang tadi dibuat
5
6 print (matrix_x) #print matrix_x yang tadi dibuat yang berbentuk
    10x10

```

```

# In[2]
import matplotlib.pyplot as KK #M
KK.plot([1,1,7,4,0,6,6]) #Memasuk
KK.xlabel('KK KMLDN') #Menambahka
KK.ylabel('1174067') #Menambahkan
KK.show() #Menampilkan grafik plo

```

[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]]

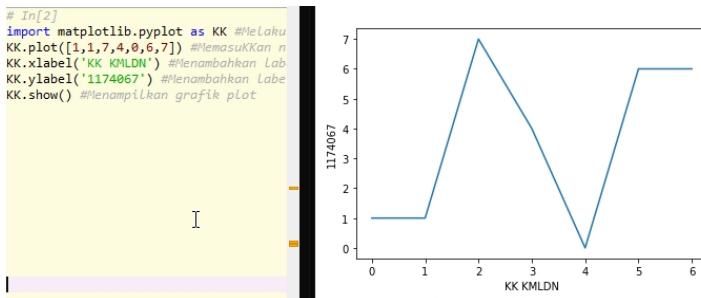
Gambar 3.5 Membuat Aplikasi pakai numpy

3.1.2.3 Nomor 3

```

1 # In [2]
2 import matplotlib.pyplot as KK #Melakukan import library numpy
    menjadi nama sendiri yaitu KK
3 KK.plot([1,1,7,4,0,6,7]) #Memasukkan nilai pada plot
4 KK.xlabel('KK KMLDN') #Menambahkan label pada x
5 KK.ylabel('1174067') #Menambahkan label pada y

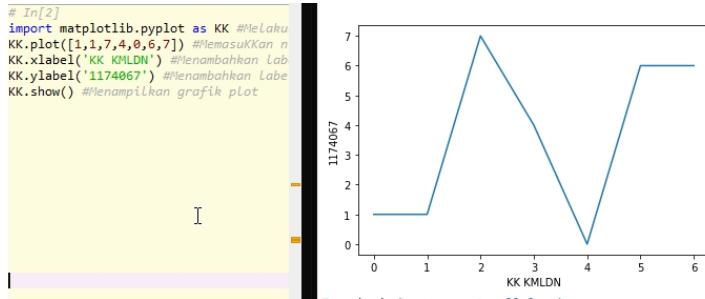
```



Gambar 3.6 Membuat Aplikasi pakai matplotlib

3.1.2.4 Nomor 4

- Source Code pertama pada random forest berfungsi untuk membaca dataset yang memiliki format text file dengan mendefinisikan variabel yang bernama imgatt. Variabel tersebut berisi value untuk membaca data.



Gambar 3.7 Hasil 4 Bagian 1

- Pada source code berikutnya akan mengembalikan baris teratas dari DataFrame variabel imgatt.

```
In [15]: import pandas as pd #Melakukan import Library numpy menjadi pd
...
...: imgatt = pd.read_csv("D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/KB3C/src/1174067/3/
CUB_2011/attributes/image_attribute_labels.txt",
...: sep='\t', header=None, error_bad_lines=False, warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present']) #membuat variable
imgatt untuk membaca file csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.8 Hasil 4 Bagian 2

- Pada output berikutnya akan menampilkan jumlah kolom dan baris dari DataFrame variable imgatt.

```
In [13]: imgatt.head() #Menampilkan 5 baris pertama
Out[13]:
      imgid  attid  present
0        1       1        0
1        1       2        0
2        1       3        0
3        1       4        0
4        1       5        1
```

Gambar 3.9 Hasil 4 Bagian 3

- Variabel imgatt2 telah menggunakan fungsi yang bernama pivot agar mengubah kolom jadi baris dan sebaliknya dari DataFrame imgatt sebelumnya.

```
1 sep='\\s+', header=None, error_bad_lines=
  False, warn_bad_lines=False,
```

In [15]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present') #Membuat sebuah variabel baru dari fungsi imgatt, dengan menggunakan index menjadi kolom dan kolom menjadi index

Gambar 3.10 Hasil 4 Bagian 4

- Variabel imgatt2 head berfungsi untuk mengembalikan value teratas pada DataFrame imgatt2.

```
1 # In [4]
```

```
In [16]: imgatt2.head() #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas
Out[16]:
attid 1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
imgid
1 0 0 0 0 1 0 0 ... 0 0 1 0 0 0 0
2 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0
3 0 0 0 0 1 0 0 ... 0 0 1 0 0 1 0
4 0 0 0 0 1 0 0 ... 1 0 0 0 1 0 0
5 0 0 0 0 1 0 0 ... 0 0 0 0 0 0 0

[5 rows x 312 columns]
```

Gambar 3.11 Hasil 4 Bagian 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

In [17]: imgatt2.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[17]: (11788, 312)

Gambar 3.12 Hasil 4 Bagian 6

- Menunjukkan dalam melakukan pivot yang mana imgid menjadi sebuah index yang unik.

```
1 imgatt2.head() #Menampilkan data yang sudah dibaca tadi tapi
  cuman data paling atas
2
3 # In [8]
4 imgatt2.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [21]: imglabels = pd.read_csv("CUB_200_2011/image_class_labels.txt",
... sep=' ', header=None, names=['imgid', 'label']) #Membaca data dimasukkan ke variable
imglabels
...: imglabels = imglabels.set_index('imgid') #Variable imglabels dan set index (imgid)
```

Gambar 3.13 Hasil 4 Bagian 7

- Akan melakukan load jawabannya yang berisi apakah burung tersebut termasuk spesies yang mana. Kolom tersebut yaitu imgid dan label.

```
1     sep=',', header=None, names=['imgid',
      'label']) #baca data csv dengan ketentuan yang ada
```

In [22]: imglabels.head()

Out[22]:

	label
imgid	
1	1
2	1
3	1
4	1
5	1

Gambar 3.14 Hasil 4 Bagian 8

- Menunjukkan bahwa jumlah baris sebanyak 11788 dan kolom 1 yang dimana kolom tersebut adalah jenis spesies pada burung.

In [23]: imglabels.shape

Out[23]: (11788, 1)

Gambar 3.15 Hasil 4 Bagian 9

- Melakukan join antara imgatt2 dengan imglabels dikarenakan memiliki isi yang sama sehingga akan mendapatkan sebuah data ciri-ciri dan data jawaban sehingga bisa dikategorikan sebagai supervised learning.

```
1     sep=',', header=None, names=['imgid',
      'label']) #Membaca data dimasuKKan ke variable imglabels
```

```
In [24]: df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi join dari data imgatt2 ke variabel imglabels
...: df = df.sample(frac=1) #Variabel df sebagai sample dengan ketentuan frac=1
```

Gambar 3.16 Hasil 4 Bagian 10

- Melakukan drop pada label yang ada didepan dan akan menggunakan label yang baru di joinkan.

```
1 # In [10]
2 imglabels.head() #Menampilkan data yang sudah dibaca tadi
      tapi cuman data paling atas
```

```
In [25]: df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
...: df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan 312
```

Gambar 3.17 Hasil 4 Bagian 11

- Mengecek isi 5 data teratas pada df.att.

```
1 imglabels.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [26]: df_att.head() #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas
Out[26]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
10397  0    0    0    0    0    0    1    ...    0    0    0    0    0    0    1
1108   0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
2382   0    0    0    0    0    0    0    ...    0    1    0    1    0    0    0
9374   0    0    0    0    0    0    1    ...    0    0    0    0    0    0    0
9371   0    0    0    0    0    0    0    ...    0    0    0    1    0    0    0
```

[5 rows x 312 columns]

Gambar 3.18 Hasil 4 Bagian 12

- Mengecek isi data teratas dari df.label.

```
1 df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi
      join dari data imgatt2 ke variabel imglabels
```

```
In [27]: df_label.head()
Out[27]:
label
imgid
10397    177
1108      20
2382      42
9374      160
9371      160
```

Gambar 3.19 Hasil 4 Bagian 13

- Membagi 8000 row pertama menjadi data training dan sisanya adalah data testing.

```
1 # In[13]
2 df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
3 df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan
312
4
5 # In[14]
6 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

```
In [29]: df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan sisanya adalah data
testing
...: df_train_label = df_label[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan sisanya adalah
data testing
...: df_test_att = df_att[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row pertama menjadi data
training dan sisanya adalah data testing
...: df_test_label = df_label[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row pertama menjadi data
training dan sisanya adalah data testing
...:
...: df_train_label = df_train_label['label'] #Menambahkan Label
...: df_test_label = df_test_label['label'] #Menambahkan Label
```

Gambar 3.20 Hasil 4 Bagian 14

- Pemanggilan class RandomForestClassifier. Dimana artinya menunjukkan banyak kolom pada setiap tree adalah 50.

```
1
2 # In[16]
```

```
In [30]: from sklearn.ensemble import RandomForestClassifier #import randomforestclassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100) #clf sebagai variabel untuk
klaifikasi random forest
```

Gambar 3.21 Hasil 4 Bagian 15

- Menunjukkan hasil prediksi dari Random Forest.

```
1 df_test_att = df_att[8000:] #Berbalik dari sebelumnya data
akan dibagi mulai dari 8000 row pertama menjadi data
training dan sisanya adalah data testing
```

```
In [31]: clf.fit(df_train_att, df_train_label) #Variablnle clf untuk fit yaitu menjadi data training
Out[31]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=None, max_features=50, 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, n_estimators=100,
n_jobs=None, oob_score=False, random_state=0, verbose=0,
warm_start=False)
```

Gambar 3.22 Hasil 4 Bagian 16

- Menampilkan besaran akurasi dari prediksi pada Random Forest yang merupakan score perolehan klarifikasi.

```
1 df_train_label = df_train_label['label'] #Menambahkan label
```

```
In [32]: print(clf.predict(df_train_att.head()))
data paling atas
[177  20  42 160 160]
```

Gambar 3.23 Hasil 4 Bagian 17

- Menampilkan besaran akurasi dari prediksi pada Random Forest yang merupakan score perolehan klarifikasi.

```
1 # In [17]
```

```
In [33]: clf.score(df_test_att, df_test_label)
Out[33]: 0.44931362196409713
```

Gambar 3.24 Hasil 4 Bagian 18

3.1.2.5 Nomor 5

```

1 clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit
   yaitu menjadi data training
2
3 # In [19]

```

```

In [34]: from sklearn.metrics import confusion_matrix #Mengimport Confusion Matrix
...: pred_labels = clf.predict(df_test_att) #membuat variable pred_Label yang data testing dari sebelumnya
...: cm = confusion_matrix(df_test_label, pred_labels) #Variable cm sebagai variabel data label
...

```

Gambar 3.25 Hasil 5 Bagian 1

```
1 # In [20]
```

```

In [35]: cm #Menampilkan data label berbentuk array
Out[35]:
array([[ 6,  0,  1, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 5,  1,  9, ...,  0,  0,  0],
       ...,
       [ 1,  0,  0, ...,  1,  0,  0],
       [ 0,  1,  0, ...,  0,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 19]], dtype=int64)

```

Gambar 3.26 Hasil 5 Bagian 2

```

1
2
3 # In [21]: Confusion Matrix
4 from sklearn.metrics import confusion_matrix #Mengimport
   Confusion Matrix
5 pred_labels = clf.predict(df_test_att) #Membuat variable
   pred_labels dari data testing
6 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
   variabel data label
7
8 # In [22]
9 cm #Memunculkan data label berbentuk array
10
11 # In [23]
12 import matplotlib.pyplot as plt #Mengimport library matplotlib
   sebagai plt
13 import itertools #Mengimport library itertools
14 def plot_confusion_matrix(cm, classes,
   normalize=False,
   title='Confusion matrix',
   cmap=plt.cm.Blues): #Membuat fungsi
   dengan ketentuan data yang ada pada cm
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

```

```
21     print("Normalized confusion matrix") #Jika normalisasi  
22     sebagai ketentuan yang ada maka print normalized confusion  
23     matrix  
24     else:  
25         print('Confusion matrix, without normalization') #Jika  
26     tidak memenuhi kondisi if maka print else  
27     print(cm) #Print data cm  
28  
29     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt  
30     sebagai fungsi untuk membuat plot  
31     plt.title(title) #Membuat title pada plot
```

```
In [36]: import matplotlib.pyplot as plt #import library matplotlib sebagai plt
...: import ertools #import library ertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues): #membuat fungsi dengan ketentuan data yang ada pada cm
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix") #jika normalisasi sebagai ketentuan yang ada maka print normalized
confusion matrix
...:     else:
...:         print("Confusion matrix, without normalization") #jika tidak memenuhi kondisi if maka print else
...:
...:     print(cm) #print data cm
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt sebagai fungsi untuk membuat plot
...:     plt.title(title) #membuat title pada plot
...:     plt.colorbar() #membuat colorbar pada plot
...:
...:     tick_marks = np.arange(len(classes)) #membuat marks pada plot
...:     plt.xticks(tick_marks, classes, rotation=90) #membuat ticks pada x
...:     plt.yticks(tick_marks, classes) #membuat ticks pada y
...:
...:     fmt = '.2f' #format sebagai normalisasi
...:     thresh = cm.max() / 2. #variabel thresh memambil data max pada cm kemudian dibagi 2
...:
...:     plt.tight_layout() #mengatur layout pada plot
...:     plt.ylabel('True label') #memberi nama Label pada sumbu y
...:     plt.xlabel('Predicted label') #memberi nama Label pada sumbu x
```

Gambar 3.27 Hasil 5 Bagian 3

```
1 plt.yticks(tick_marks, classes) #Membuat ticks pada y
2
3 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
4 thresh = cm.max() / 2. #Variable thresh menambil data max
   pada cm kemudian dibagi 2
```

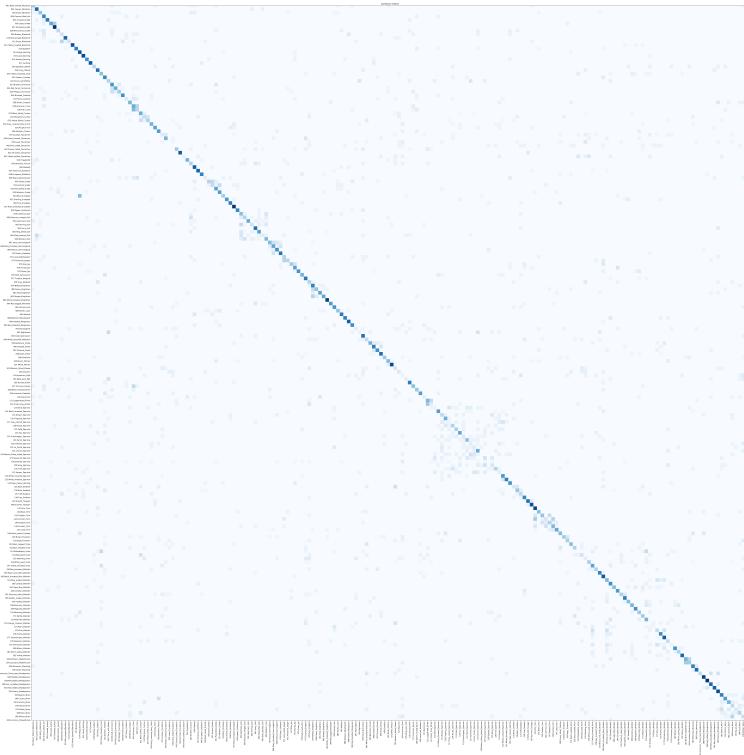
```
In [37]: birds = pd.read_csv("CUB_200_2011/cla
...:                                     sep='\s+', header=
birdname
...: birds = birds['birdname'] #nama birds
...: birds #menampilkan data birds
Out[37]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195      196.House_Wren
196      197.Marsh_Wren
197      198.Rock_Wren
198      199.Winter_Wren
199      200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

Gambar 3.28 Hasil 5 Bagian 4

```
1 plt.xlabel('Predicted label') #Menambahkan nama label pada
2 sumbu x
3
4 # In [24]
5 birds = pd.read_csv("N:/Tugas/Kuliah/Semester 6/Kecerdasan Buatan
6 /KB3C Ngerjain/src/1174067/3/CUB_200_2011/classes.txt",
7                                     sep='\s+', header=None, usecols=[1], names=['
birdname']) #membaca csv dengan ketentuan nama birdname
```

```
In [39]: import numpy as np #import library numpy sebagai np
...: np.set_printoptions(precision=2) #np sebagai variabel yang membuat set precision=2
...: plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure dengan ketentuan size 60,60 dan dpi 300
...: plot_confusion_matrix(cm, classes=birds, normalize=True) #data cm dan clas birds dibuat sebagai plot
...: plt.savefig('hasil.png')
Normalized confusion matrix
[[0.37 0. 0. ... 0. 0.05 0. ]
 [0.08 0.77 0. ... 0. 0. 0. ]
 [0.04 0.04 0.39 ... 0. 0. 0. ]
 ...
 [0. 0. 0. ... 0.31 0. 0. ]
 [0. 0. 0. ... 0. 0.24 0. ]
 [0. 0. 0. ... 0. 0.04 0.73]]
```

Gambar 3.29 Hasil 5 Bagian 5



Gambar 3.30 Plot Hasil 5 Bagian 5

3.1.2.6 Nomor 6

```

1 # In [25]
2 import numpy as np #Mengimport library numpy sebagai np
3 np.set_printoptions(precision=2) #np sebagai variabel yang
   membuat set precision=2
4 plt.figure(figsize=(60,60), dpi=300) #Plot sebagai figure dengan
   ketentuan size 60,60 dan dpi 300

```

```

In [24]: from sklearn import tree #Mengimport library
....: clftree = tree.DecisionTreeClassifier() #clf
....: clftree.fit(df_train_att, df_train_label) #M
....: clftree.score(df_test_att, df_test_label) #M
Out[24]: 0.264519535374868

```

Gambar 3.31 Hasil 6 Bagian 1

```

1 #plt.savefig('hasil.png') #

```

```
In [31]: from sklearn.model_selection import
...: scores = cross_val_score(clf, df_dari_data_training
...: print("Accuracy: %0.2f (+/- %0.2f"
C:\ProgramData\Anaconda3\lib\site-packages\from 'auto' to 'scale' in version 0.22 to ; avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.27 (+/- 0.01)
```

Gambar 3.32 Hasil 6 Bagian 2

3.1.2.7 Nomor 7

```
1 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
   untuk decision tree
2 clftree.fit(df_train_att, df_train_label) #Mengatur data training
3 clftree.score(df_test_att, df_test_label) #Mengatur data testing
```

```
In [32]: scorestree = cross_val_scores dan metode tree
...: print("Accuracy: %0.2f ada
Accuracy: 0.44 (+/- 0.02)
```

Gambar 3.33 Hasil 7 Bagian 1

```
1 from sklearn import svm #Mengimport library svm
2 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur fungsi SVC
```

In [28]: scorestree = cross_val_scores dan metode tree
: print("Accuracy: %0.2f ada
 Accuracy: 0.27 (+/- 0.02)

Gambar 3.34 Hasil 7 Bagian 2

1 # In [28]: Pengecekan Cross Validation

```
In [27]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5) #Membuat variable data training
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2)) #Menampilkan data testing dan output akurasi
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)
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)
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)
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)
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)
Accuracy: 0.27 (+/- 0.03)
```

Gambar 3.35 Hasil 7 Bagian 3

3.1.2.8 Nomor 8

```
1 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
() * 2)) #Print data scores dengan ketentuan akurasi
2
3 # In [29]
4 scorestree = cross_val_score(clftree, df_train_att,
    df_train_label, cv=5) #Membuat variable prediksi menggunakan
    scores dan metode tree
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
    scorestree.std() * 2)) #Menampilkan dengan ketentuan yang ada
6
7 # In [30]
8 scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label,
    cv=5) #Membuat variable data training
9 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
    scoressvm.std() * 2)) #Menampilkan data testing dan output
    akurasi
```

```

11 # In [31]: Pengamatan komponen informasi
12 max_features_opts = range(5, 50, 5) #Variatele max_features_opts
   sebagai variabel untuk membuat range 5,50,5
13 n_estimators_opts = range(10, 200, 20) #Variablen_estimators_opts
   sebagai variabel untuk membuat range 10,200,20
14 rf_params = np.empty((len(max_features_opts)*len(
   n_estimators_opts),4), float) #Variablerf_params sebagai
   variabel untuk menjumlahkan yang sudah di tentukan sebelumnya
15 i = 0
16 for max_features in max_features_opts: #Perulangan

```

```

....          ..._PENGAMATAN_KOMPONENTE_INFORMASI
....      rf_params[i,3] = scores.std() * 2 #index 3
....      i += 1 #Dengan ketentuan i += 1
....      print("Max features: %d, num estimators: %d, acc
n_estimators, scores.mean(), scores.std() * 2))
....      #Print hasil pengulangan yang sudah ditentukan
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.03)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 190, accuracy: 0.45 (+/- 0.01)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.02)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.02)
Max features: 10, num estimators: 50, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.31 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 15, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)

```

Gambar 3.36 Hasil 8 Bagian 1

```

Max features: 40, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 10, accuracy: 0.34 (+/- 0.02)
Max features: 45, num estimators: 30, accuracy: 0.41 (+/- 0.02)
Max features: 45, num estimators: 50, accuracy: 0.43 (+/- 0.02)
Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 110, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 190, accuracy: 0.45 (+/- 0.02)

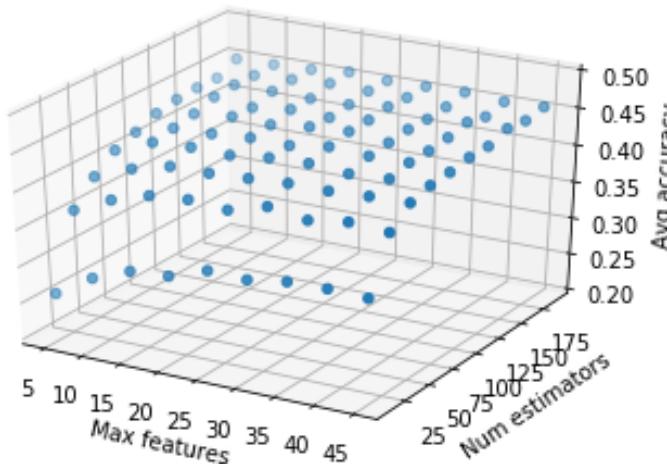
```

Gambar 3.37 Hasil 8 Bagian 1 Akhir kode

```

1      clf = RandomForestClassifier(max_features=max_features ,
2                                     n_estimators=n_estimators) #Menampilkan variabel csf
3                                     scores = cross_val_score(clf , df_train_att ,
4                                     df_train_label , cv=5) #Variable scores sebagai variabel
5                                     training
6                                     rf_params[i,0] = max_features #index 0
7                                     rf_params[i,1] = n_estimators #index 1
8                                     rf_params[i,2] = scores.mean() #index 2
9                                     rf_params[i,3] = scores.std() * 2 #index 3
10                                    i += 1 #Dengan ketentuan i += 1
11                                    print("Max features: %d, num estimators: %d, accuracy:
12                                     %0.2f (+/- %0.2f)" % (max_features , n_estimators ,
13                                     scores.mean() , scores.std() * 2))
14                                     #Print hasil pengulangan yang sudah ditentukan
15
# In [32]
import matplotlib.pyplot as plt #Mengimport library matplotlib
sebagai plt
from mpl_toolkits.mplot3d import Axes3D #Mengimport axes3D untuk
menampilkan plot 3 dimensi
from matplotlib import cm #Memanggil data cm yang sudah tersedia
fig = plt.figure() #Menghasilkan plot sebagai figure

```



Gambar 3.38 Hasil 8 Bagian 2

3.1.3 Penanganan Error

1. ScreenShoot Error

FileNotFoundError: [Errno 2] File 'N:\Data\Iris\IrisTest.csv' not found: 'C:\Users\Acer\PycharmProjects\Klasifikasi\IrisTest.py': line 10: syntax error near unexpected token `('

Gambar 3.39 FileNotFoundError

2. Tuliskan Kode Error dan Jenis Error

- FileNotFoundError

3. Cara Penangan Error

- FileNotFoundError

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

3.1.4 Link Youtube:

https://youtu.be/69BWGlcf_CQ

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.

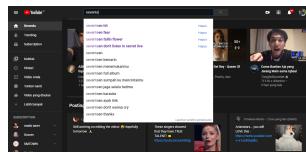
Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang 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 :



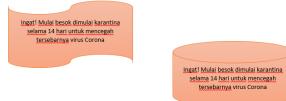
Gambar 4.5 Klasifikasi teks Youtube.

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

```

```
1 dvec #Untuk menampilkan hasil dari code sebelumnya
2
3 Daptarkata= vectorizer.get_feature_names()
4
5 dshuf = fanny.sample(frac=1)
6
7
8 d_train=dshuf[:300]
9 d_test=dshuf[300:]
10
11 d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
12 d_train_att
13
14 d_train_label=d_train['CLASS']
15 d_test_label=d_test['CLASS']
```

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

Fancy - Database					
Index	COMMAND_ID	AUTHOR	DATE	CONTENT	CLASS
0	133hex00fond	Corey Wiley	2015-05-28T21:00:00	a=HTTP://... wired but funny	Rock
1	134hex00fond	Tori Sweng	2015-05-28T21:00:00	wired, goss, I am a hu...	Rock
2	135hex00fond	Lad Music	2015-05-28T21:00:00	lads, rock, lol...	Rock
3	136hex00fond	Cheryl Fox	2015-05-28T21:00:00	party rock	Party rock
4	137hex00fond	PATRIC_TW	2015-05-28T21:00:00		
5	138hex00fond	Brian Brat	2015-05-28T21:00:00	Shurts	Rock
6	139hex00fond	BRADY	2015-05-28T21:00:00	One song	Rock
7	1310hex00fond	Alex Defeo	2015-05-28T21:00:00	one song is really	Rock
8	1311hex00fond	Jlement	2015-05-28T21:00:00	Awesomebr />	Rock
9	1312hex00fond	Silvia Russo	2015-05-28T21:00:00	silvia is reliable inc	Rock
10	1320hex00fond	Michael	2015-05-28T21:00:00	I love this	Rock
11	1321hex00fond	jerri merris	2015-05-28T21:00:00	jerri merris	Rock
12	1322hex00fond	Alex Martin	2015-05-28T21:00:00	80's Litter	Rock
13	1323hex00fond	Alex Jansen	2015-05-28T21:00:00	i kiss shes	Rock
14	1324hex00fond	Aiden Hill	2015-05-27T22:00:00	the best	Rock
15	1325hex00fond	Joe pepin	2015-05-27T22:00:00	Best song	Rock
16	1326hex00fond	John Morrison	2015-05-27T22:00:00	sugar slice,	Rock
17	1327hex00fond	hilal koren	2015-05-27T22:00:00	the mouse	Rock
18	1328hex00fond		2015-05-27T22:00:00	wwwwww...	Rock
19	1329hex00fond	SmashDad	2015-05-27T22:00:00	PARTY ROCK	Party rock
20	1330hex00fond	Phuong Linh	2015-05-27T22:00:00	you wake me up	Rock
21	1331hex00fond	Nguyen Ngoc	2015-05-27T22:00:00	you watched -	Rock
22	1332hex00fond	Ferrare	2015-05-27T22:00:00	(increased)	Rock
23	1333hex00fond	louise taylor	2015-05-27T22:00:00	you're	Rock

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, coef0=0.0, degree=3, epsilon=0.1,
gamma='auto',
      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
17 : %.2f (+-%.2f)" % (max_features, n_estimators, scores.
mean(), scores.std() * 2))
    #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 :

Gambar 4.15 Hasil Soal 8.

4.2.3 Penanganan Error

1. ScreenShoot Error

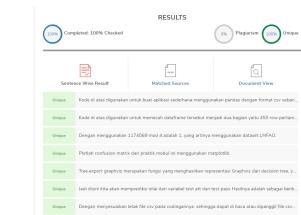
```
FileNotFoundException: [Errno 2] File b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/fanny.csv' does not exist: b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/fanny.csv'
```

2. Cara Penangan 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 :



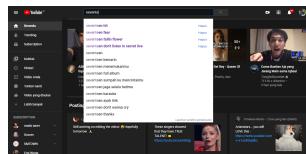
Gambar 4.18 Klasifikasi teks.

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.

- 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.

- 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.
- 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.

- 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 :

Genre - DataName						
Index	Comment_ID	AUTHOR	DATE	CONTENT	CLASS	
0	113wunfghd	Cory Wilson	2010-05-28T12:00:00	here's a href="http://www.	+	
1	11342crxh3bd	Talia Goming	2010-05-28T12:00:00	funny	+	
2	113t3yjy1mz	Luis Munic	2010-05-28T12:00:00	that's it! I'm a noob	+	
3	113t10nq3mz	Ferney Fox	2010-05-28T12:00:00	Rock...lol :)	+	
4	113p0c14sed	PATRICKU	2010-05-28T12:00:00	Parry Rock	+	
5	113t3yjy1mz	Brian Brat	2010-05-28T12:00:00	Shuttle	+	
6	113t3el5l0oy	Brian Brat	2010-05-28T12:00:00	Ong	+	
7	113t300000000000	Alex Defer	2010-05-28T12:00:00	This song is	+	
8	113t3yjy1mz	Giovanni	2010-05-28T12:00:00	so real	+	
9	113t3yjy1mz	Silvia Bresser	2010-05-28T12:00:00	Awesome :)	+	
10	113t3yjy1mz	Himawari	2010-05-28T12:00:00	I love this	+	
11	113t3yjy1mz	Yuri mafra	2010-05-28T12:00:00	song	+	
12	113t3yjy1mz	Alvaro	2010-05-28T12:00:00	2011 LIEEEF	+	
13	113t3yjy1mz	Alex Martin	2010-05-28T12:00:00	i add this	+	
14	113t3yjy1mz	people dress	2010-05-28T12:00:00	up	+	
15	113t3yjy1mz	Alex Jansen	2010-05-28T12:00:00	last year of	+	
16	113t3yjy1mz	alexander willi	2010-05-27T22:00:00	ever!!!!!!	+	
17	113t3yjy1mz	peopl...	2010-05-27T22:00:00	for me,	+	
18	113t3yjy1mz	lebanic	2010-05-27T22:00:00	love music	+	
19	113t3yjy1mz	luha kerino	2010-05-27T22:00:00	down	+	
20	113t3yjy1mz	SoulInCloud	2010-05-27T22:00:00	(8)	+	
21	113t3yjy1mz	lucy	2010-05-27T22:00:00	you're so	+	
22	113t3yjy1mz	Nguyen Ngoc	2010-05-27T22:00:00	it's	+	
23	113t3yjy1mz	lucy	2010-05-27T22:00:00	you watched	+	
24	113t3yjy1mz	lucy	2010-05-27T22:00:00	the	+	
25	113t3yjy1mz	lucy	2010-05-27T22:00:00	incredible!	+	
26	113t3yjy1mz	lucy	2010-05-27T22:00:00	you	+	
27	113t3yjy1mz	lucy	2010-05-27T22:00:00	is	+	
28	113t3yjy1mz	lucy	2010-05-27T22:00:00	you're	+	
29	113t3yjy1mz	lucy	2010-05-27T22:00:00	so	+	
30	113t3yjy1mz	lucy	2010-05-27T22:00:00	amazing	+	
Format	Reset	Background color	Column menu			

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(dtrain_att, dtrain_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 :

Gambar 4.30 Hasil Soal 8.

4.3.3 Penanganan Error

1. ScreenShoot Error

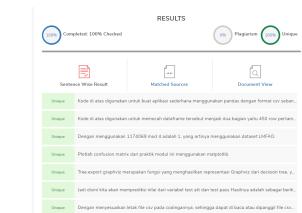
```
FileNotFoundException: [Errno 2] File b'F:\\Semester 6\\Artificial Intelligence\\Tugas 4\\src\\fanny.csv' does not exist: b'F:\\Semester 6\\Artificial Intelligence\\Tugas 4\\src\\fanny.csv'
```

2. Cara Penangan 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

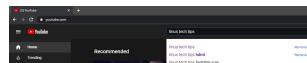
Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang 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:



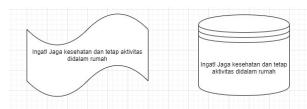
Gambar 4.35 Klasifikasi Teks pada Youtube.

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

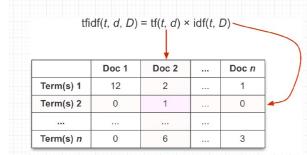
Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut:



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:



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

```

```

In [42]: d_train_label = d_train['CLASS']
        d_test_label = d_test['CLASS']

In [43]: d_train_att = d_train.drop(['CLASS'], axis=1)
        d_test_att = d_test.drop(['CLASS'], axis=1)

In [44]: d_train_att.info()
Out[44]:

RangeIndex: 395 entries, 0 to 395
Data columns (total 10 columns):
 #   Column  Non-Null Count  Dtype  
--- 
 0   ID       395 non-null   int64  
 1   SURVIVED 395 non-null   int64  
 2   Pclass    395 non-null   int64  
 3   Name     395 non-null   object 
 4   Sex      395 non-null   object 
 5   Age      395 non-null   float64
 6   SibSp    395 non-null   int64  
 7   Parch    395 non-null   int64  
 8   Fare     395 non-null   float64
 9   Cabin    395 non-null   object 
dtypes: float64(2), int64(5), object(3)
memory usage: 33.9 KB
In [45]: d_train_att.describe()
Out[45]:
          ID         Pclass      Age      SibSp      Parch      Fare
count  395.000000  395.000000  395.000000  395.000000  395.000000  395.000000
mean   197.000000   3.333333  30.092593   0.538462   0.416667  32.204724
std    149.015157   1.712368  16.535057   0.834922   0.834922  40.190432
min    1.000000   1.000000  0.000000   0.000000   0.000000  0.000000
25%    46.000000   2.000000  22.000000   0.000000   0.000000  7.250000
50%    147.500000   3.000000  27.500000   0.000000   0.000000  14.500000
75%    247.500000   4.000000  38.000000   1.000000   1.000000  31.200000
max    395.000000   6.000000  80.000000   6.000000   6.000000  512.000000

```

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

```

```

In [46]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
C:\Users\Naufal\PycharmProjects\Klasifikasi\Pelajaran: The default value of gamma will change
from 'scale' to 'auto' in version 0.20 to account better for sparse features. Set gamma explicitly to 'scale' or 'auto' to
avoid this warning.
UserWarning: gamma is not specified, 'auto' is selected.
Out[46]: 0.9649594594594594

```

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

```

```

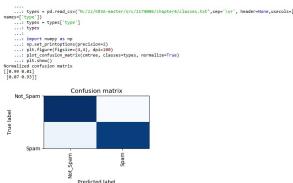
In [45]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
...: clftree.score(d_test_att, d_test_label)
Out[45]: 0.9594594594594594

```

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 scoressvm = cross_val_score(clfsvm, d_train_att, d_train_label,
9                             cv=5)
10 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
11                                         scoressvm.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 [30]: 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))
... scoressvm = cross_val_score(clfsvm, d_train_att, d_train_label, cv=5)
... print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
...                                         scoressvm.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: 0.97 (+/- 0.00)
Accuracy: 0.97 (+/- 0.00)

In [31]:
```

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 = 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:  
%0.2f (+/- %0.2f)" % (max_features, n_estimators, scores.mean()  
(), scores.std() * 2))
```

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] File 'B\N\Tugas\a\Semester 6\Kecerdasan Buatan\KB3C
| Ngerjan\src\11740664\csv.csv' does not exist: 'B\N\Tugas\a\Semester 6\Kecerdasan
| Buatan\KB3C\Ngerjan\src\11740664\csv.csv'
```

Gambar 4.46 FileNotFoundErrors

2. Cara Penangan 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.

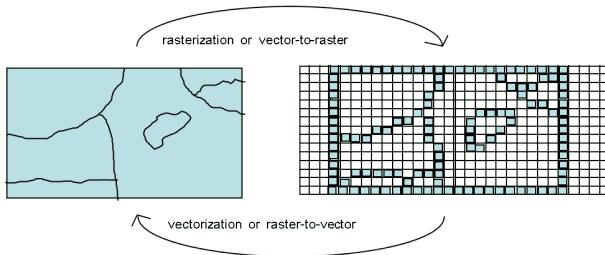
Menggunakan teknik bag-of-words pada klasifikasi berbasis text dan kata untuk mengklasifikasikan komentar yang ada di internet sebagai spam atau bukan. Misalkan pada kolom komentar dapat di cek seberapa sering suatu kata muncul dalam kalimat. Setiap kata dapat dijadikan baris dan kolomnya ini merupakan 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 sebanyak berapa kali 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	green eyes	black hair
487	Keniuchio	Public	Neutral	Brown Eyes	Black Hair
488	Harada (Earth-616)	Identity	Characters		
489	Irene Adler (Earth-616)	Secret	Neutral	White Eyes	Silver Hair
490	Marrina (Earth-616)	Identity	Good	Black Eyes	Green Hair
491	Smallwood (Earth-616)	Identity	Characters		
492	Fred Davis Jr. (Earth-616)	Secret	Good	Blue Eyes	White Hair
493	Zelda DuBois (Earth-616)	Public	Bad	Green Eyes	Auburn Hair
494	Beyonder (Earth-616)	Identity	Neutral		
495	Roxanne Washington (Earth-616)	Secret	Characters	Variable Eyes	Variable Hair
496	Bonita Juarez (Earth-616)	Identity	nan	nan	nan
497	Surtur (Earth-616)	No Dual Identity	Good	Brown Eyes	Black Hair
498	Stranger (Cosmic Being)	Identity	Bad	Yellow Eyes	Red Hair
499	Brandy Clark (Earth-616)	Public	Neutral	White Eyes	White Hair
	Lillian Crawley (Earth-616)	Secret	Characters	nan	Brown Hair
	Ichabod (Earth-616)	Identity	Good	Hazel Eyes	Brown Hair
			Characters	nan	White Hair

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.

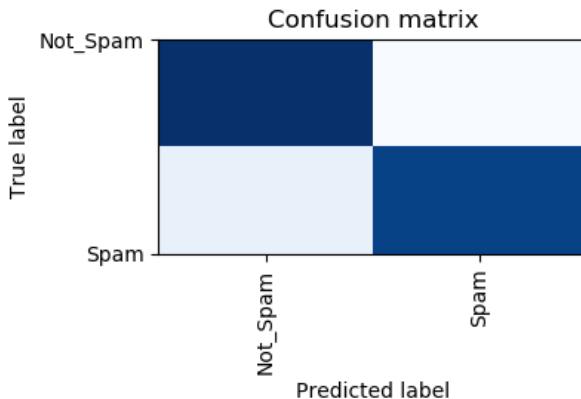
```
1 from sklearn.metrics import confusion_matrix
2 pred_labeltree = clftree.predict(d_test_att)
3 cmmtree = confusion_matrix(d_test_label, pred_labeltree)
4 cmmtree
5
6 import matplotlib.pyplot as plt
7 import itertools
8 def plot_confusion_matrix(cm, classes,
9                         normalize=False,
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)

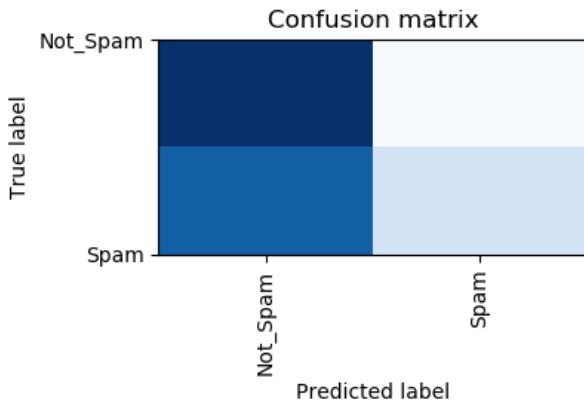
```

```

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 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(cmsvm, classes=types, normalize=True)
53 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.

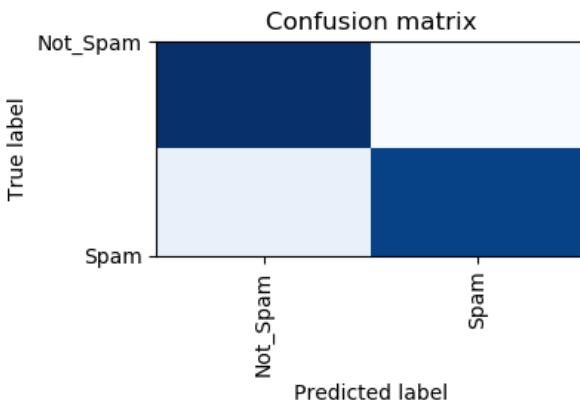
```

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=[1], names=['type'])
42 types = types['type']
43 types
44
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 praktik no. 6(3)



Gambar 4.62 hasil praktik 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)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
6 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)

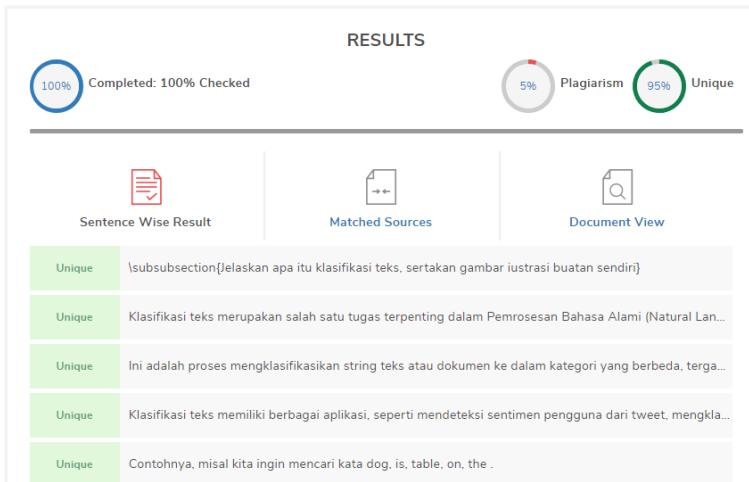
```
Max features: 25, num estimators: 150, accuracy: 0.93 (+/- 0.04)
Max features: 25, num estimators: 170, accuracy: 0.94 (+/- 0.03)
Max features: 25, num estimators: 190, accuracy: 0.94 (+/- 0.03)
Max features: 30, num estimators: 10, accuracy: 0.93 (+/- 0.06)
Max features: 30, num estimators: 30, 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 (+/- 0.05)
Max features: 45, num estimators: 170, accuracy: 0.93 (+/- 0.04)
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 Vektorisai 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)
7 # In[15]:
8 clftree.score(dk_test_att, dk_test_label) #Mengatur data
9         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, 0, 1,
       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 Desicon 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

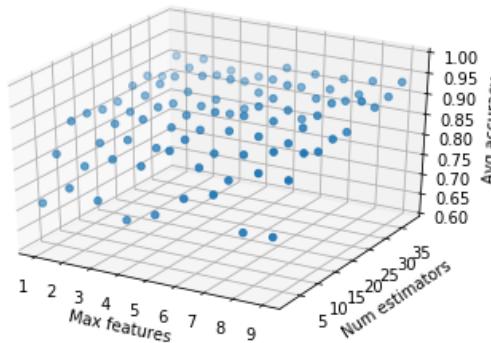
8. No. 8

```

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=
      max_features, n_estimators=n_estimators) #Menampilkan
      variabel csf
11     scores = cross_val_score(clf, dk_train_att,
      dk_train_label, cv=5) #Variable scores sebagai variabel
      training
12     rf_params[i,0] = max_features #index 0
13     rf_params[i,1] = n_estimators #index 1
14     rf_params[i,2] = scores.mean() #index 2
15     rf_params[i,3] = scores.std() * 2 #index 3
16     i += 1 #Dengan ketentuan i += 1
17     print("Max features: %d, num estimators: %d, accuracy
      : %0.2f (+/- %0.2f)")
18 % (max_features, n_estimators, scores.mean(), scores.std() *
      2)) #Print hasil pengulangan yang sudah ditentukan
19
20 # In [20]:
21 import matplotlib.pyplot as plt #Mengimport library
  matplotlib sebagai plt
22 from mpl_toolkits.mplot3d import Axes3D #Mengimport axes3D
  untuk menampilkan plot 3 dimensi
23 from matplotlib import cm #Memanggil data cm yang sudah
  tersedia
24 fig = plt.figure() #Menghasilkan plot sebagai figure
25 fig.clf() #Figure di ambil dari clf
26 ax = fig.gca(projection='3d') #ax sebagai projection 3d
27 x = rf_params[:,0] #x sebagai index 0
28 y = rf_params[:,1] #y sebagai index 1
29 z = rf_params[:,2] #z sebagai index 2
30 ax.scatter(x, y, z) #Membuat plot scatter x y z
31 ax.set_zlim(0.6, 1) #Set zlim dengan ketentuan yang ada
32 ax.set_xlabel('Max features') #Memberikan nama label x
33 ax.set_ylabel('Num estimators') #Memberikan nama label y
34 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

- FileNotFoundError

3. Cara Penanganan Error

- FileNotFoundError

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 :



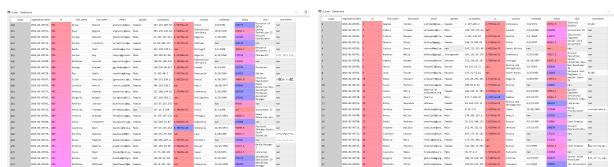
Gambar 4.87 Hasil Soal 1.

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 :

Fancy - DataName						
Index	COMMENT_ID	AUTHOR	DATE	CONTENT	CLASS	
0	1	l3huw0hpepd...	Cory Wilson	2015-05-28T17:45:00	here	/
1	124243rc4hsd...	Talit Gamed	2015-05-28T17:45:00	funny	/	
2	13t3tcjy...	Luis Musico	2015-05-28T17:45:00	funny	/	
3	13lt7WQh...nq...	Cheney Fox	2015-05-28T17:45:00	funny...lol...	/	
4	13t3pc0d...	PATRICKZU	2015-05-28T17:45:00	Party rock	/	
5	13t3ry0m...	Brian Brat	2015-05-28T17:45:00	Shuttle	/	
6	13t36el1...	Brian Brat	2015-05-28T17:45:00	ong	/	
7	13t33100...	Alex Defer	2015-05-28T17:45:00	This song is	/	
8	13t3gk...	Giovanni	2015-05-28T17:45:00	funny	/	
9	13t3zrxm...	Silvia Rescor	2015-05-28T17:45:00	Awesome!!!	/	
10	13t3c7y...	Michael	2015-05-28T17:45:00	I love this	/	
11	13t3g4et...	merly mera	2015-05-28T17:45:00	funny	/	
12	13t3t0...	Korriged	2015-05-28T17:45:00	adss when	/	
13	13t3t0...	...	2015-05-28T17:45:00	people dress	/	
14	13t3t0217...	Alex Jasee	2015-05-28T17:45:00	last year of	/	
15	13t3x0...	adam will	2015-05-28T17:45:00	ever!!!!!!	/	
16	13t3tqj3...	lebanic	2015-05-28T17:45:00	love music	/	
17	13t3ge0...	...	2015-05-27T17:45:00	soooooommmmm	/	
18	13t3fmg...	SoulDinobots	2015-05-27T17:45:00	(8)	/	
19	13t3yq...	Nguyen	2015-05-27T17:45:00	you wanted	/	
20	13t3mf...	Patagonia	2015-05-27T17:45:00	this is	/	
21	13t3wq...	...	2015-05-27T17:45:00	LMFAO!!!!!!	/	

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 :

Gambar 4.94 Hasil Soal 8.

4.7.3 Penanganan Error

1. ScreenShoot Error

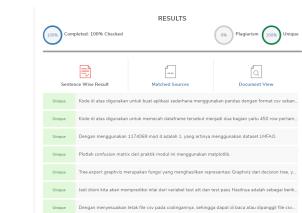
```
FileNotFoundException: [Errno 2] File b'C:/Users/ASUS/videos/raglur/snc/1174877/4/alva.csv' does  
not exist: b'C:/Users/ASUS/videos/raglur/snc/1174877/4/alva.csv'
```

2. Cara Penangan 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

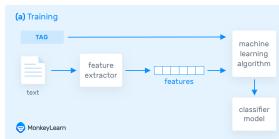
4.7.5 Link Youtube

4.8 1174079 -Chandra Kirana Poetra

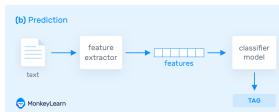
4.8.1 Teori

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
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_test_label = d_test['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 dalam 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

```

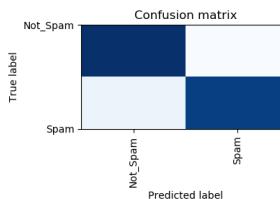
1 # Confusion Matrix – Decission Tree
2 from sklearn.metrics import confusion_matrix
3 pred_labeltree = clftree.predict(d_test_att)
4 cmtree = confusion_matrix(d_test_label, pred_labeltree)
5 cmtree
6
7 import matplotlib.pyplot as plt
8 import itertools
9 def plot_confusion_matrix(cm, classes,
                           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\cmnbase.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...]
      --help          display help and exit
      --version       display version and exit
      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 (+/- 0.06)
      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/cv_chandra.csv' from
Semester 6/kecerdasan Buata/Github/upload 30 Maret 2020 github
tugas 4/src/1174087/4/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/1174087/4/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

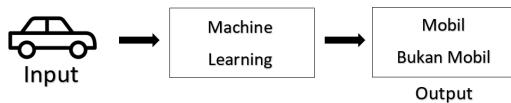
<https://youtu.be/8vkMpgUVTQs>

4.9 1174087 - Ilham Muhammad Ariq

4.9.1 Teori

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

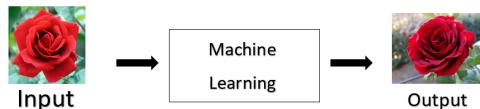
Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk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

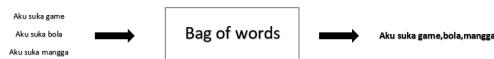
- COMMENT_ID : merupakan key unik yang membedakan komen lainnya.
- AUTHOR : merupakan penulis dari komen tersebut.
- DATE : merupakan waktu dari komen tersebut dipublikasikan.
- CONTENT : merupakan isi komentarnya.
- CLASS : merupakan klasifikasi dari komennya (spam/notspam).

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' ]
23
24

```

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

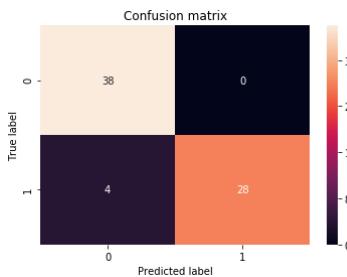
```
In [151]: from sklearn import tree
.....
Out[151]:
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_weight_fraction_leaf=0.0, presort=False,
random_state=None,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

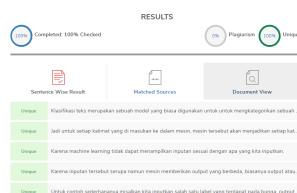
Gambar 4.125 FileNotFoundError

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

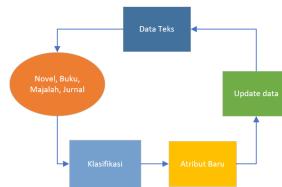
<https://youtu.be/lLxUddy2kW4>

4.10 1174086 - Tia Nur candida

4.10.1 Teori

1. Klasifikasi Text

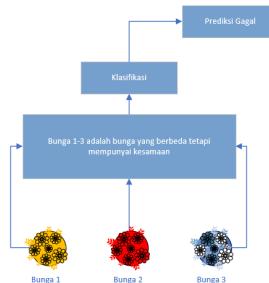
Klasifikasi teks merupakan cara dalam memilah milah data teks berdasarkan parameter tertentu dengan cara data yang bersifat dokumen ataupun teks yang memiliki kumpulan teks didalamnya, serta teks itu sendiri bertipe char atau string yang mudah diolah.



Gambar 4.127 Klasifikasi Text

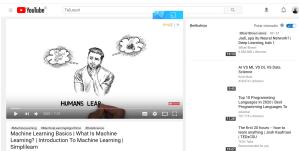
2. Mengapa Klasifikasi Bunga tidak dapat menggunakan Machine Learning

Karena klasifikasinya menggunakan tipe data yang dimana attributnya 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 2 dikarenakan memiliki perbedaan yang hampir tidak dapat dilihat dengan 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

Term	Document 1	Document 2
aid	0 1	
back	1 0	
brown	1 0	
come	1 0	
for	1 0	1 0
fox	1 0	
good	1 0	
lazy	1 0	
men	1 0	
over	0 1	
party	1 0	
their	1 0	
time	0 1	

Stopword List

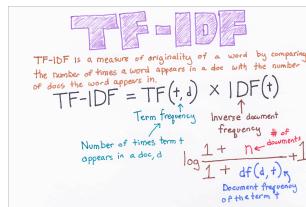
for
of
the
to

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

- import data pandas as pd #mengimport library pandas dan menamainya pd

```
1 import pandas as pd #mengimport library pandas dan menamainya
pd
2 #%%
```

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

Name	Type	Size	Value	
Column names: id, first_name, last_name, email, gender				
tia	Dataframe	(500, 5)	Column names: id, first_name, last_name, email, gender	
<hr/>				
1	Angel	Bentje	abentje@wpisquido.com	Female
2	Karen	Tokan	ktkan@wpisquido.de	Female
3	Odile	Koenig	okoenig@wpisquido.de	Female
4	Callie	Riley	criley@wpisquido.com	Female
5	Gert	Goedhart	ggoedhart@wpisquido.com	Female

Gambar 4.132 Data Dummy

- 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 recod 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] "katy"
#> [3] "katy"
#> [4] "katy"
#> [5] "katy"
#> [6] "katy"
#> [7] "katy"
#> [8] "katy"
#> [9] "katy"
#> [10] "katy"
#> [11] "katy"
#> [12] "katy"
#> [13] "katy"
#> [14] "katy"
#> [15] "katy"
#> [16] "katy"
#> [17] "katy"
#> [18] "katy"
#> [19] "katy"
#> [20] "katy"
#> [21] "katy"
#> [22] "katy"
#> [23] "katy"
#> [24] "katy"
#> [25] "katy"
#> [26] "katy"
#> [27] "katy"
#> [28] "katy"
#> [29] "katy"
#> [30] "katy"
#> [31] "katy"
#> [32] "katy"
#> [33] "katy"
#> [34] "katy"
#> [35] "katy"
#> [36] "katy"
#> [37] "katy"
#> [38] "katy"
#> [39] "katy"
#> [40] "katy"
#> [41] "katy"
#> [42] "katy"
#> [43] "katy"
#> [44] "katy"
#> [45] "katy"
#> [46] "katy"
#> [47] "katy"
#> [48] "katy"
#> [49] "katy"
#> [50] "katy"
#> [51] "katy"
#> [52] "katy"
#> [53] "katy"
#> [54] "katy"
#> [55] "katy"
#> [56] "katy"
#> [57] "katy"
#> [58] "katy"
#> [59] "katy"
#> [60] "katy"
#> [61] "katy"
#> [62] "katy"
#> [63] "katy"
#> [64] "katy"
#> [65] "katy"
#> [66] "katy"
#> [67] "katy"
#> [68] "katy"
#> [69] "katy"
#> [70] "katy"
#> [71] "katy"
#> [72] "katy"
#> [73] "katy"
#> [74] "katy"
#> [75] "katy"
#> [76] "katy"
#> [77] "katy"
#> [78] "katy"
#> [79] "katy"
#> [80] "katy"
#> [81] "katy"
#> [82] "katy"
#> [83] "katy"
#> [84] "katy"
#> [85] "katy"
#> [86] "katy"
#> [87] "katy"
#> [88] "katy"
#> [89] "katy"
#> [90] "katy"
#> [91] "katy"
#> [92] "katy"
#> [93] "katy"
#> [94] "katy"
#> [95] "katy"
#> [96] "katy"
#> [97] "katy"
#> [98] "katy"
#> [99] "katy"
#> [100] "katy"
#> [101] "katy"
#> [102] "katy"
#> [103] "katy"
#> [104] "katy"
#> [105] "katy"
#> [106] "katy"
#> [107] "katy"
#> [108] "katy"
#> [109] "katy"
#> [110] "katy"
#> [111] "katy"
#> [112] "katy"
#> [113] "katy"
#> [114] "katy"
#> [115] "katy"
#> [116] "katy"
#> [117] "katy"
#> [118] "katy"
#> [119] "katy"
#> [120] "katy"
#> [121] "katy"
#> [122] "katy"
#> [123] "katy"
#> [124] "katy"
#> [125] "katy"
#> [126] "katy"
#> [127] "katy"
#> [128] "katy"
#> [129] "katy"
#> [130] "katy"
#> [131] "katy"
#> [132] "katy"
#> [133] "katy"
#> [134] "katy"
#> [135] "katy"
#> [136] "katy"
#> [137] "katy"
#> [138] "katy"
#> [139] "katy"
#> [140] "katy"
#> [141] "katy"
#> [142] "katy"
#> [143] "katy"
#> [144] "katy"
#> [145] "katy"
#> [146] "katy"
#> [147] "katy"
#> [148] "katy"
#> [149] "katy"
#> [150] "katy"
#> [151] "katy"
#> [152] "katy"
#> [153] "katy"
#> [154] "katy"
#> [155] "katy"
#> [156] "katy"
#> [157] "katy"
#> [158] "katy"
#> [159] "katy"
#> [160] "katy"
#> [161] "katy"
#> [162] "katy"
#> [163] "katy"
#> [164] "katy"
#> [165] "katy"
#> [166] "katy"
#> [167] "katy"
#> [168] "katy"
#> [169] "katy"
#> [170] "katy"
#> [171] "katy"
#> [172] "katy"
#> [173] "katy"
#> [174] "katy"
#> [175] "katy"
#> [176] "katy"
#> [177] "katy"
#> [178] "katy"
#> [179] "katy"
#> [180] "katy"
#> [181] "katy"
#> [182] "katy"
#> [183] "katy"
#> [184] "katy"
#> [185] "katy"
#> [186] "katy"
#> [187] "katy"
#> [188] "katy"
#> [189] "katy"
#> [190] "katy"
#> [191] "katy"
#> [192] "katy"
#> [193] "katy"
#> [194] "katy"
#> [195] "katy"
#> [196] "katy"
#> [197] "katy"
#> [198] "katy"
#> [199] "katy"
#> [200] "katy"
#> [201] "katy"
#> [202] "katy"
#> [203] "katy"
#> [204] "katy"
#> [205] "katy"
#> [206] "katy"
#> [207] "katy"
#> [208] "katy"
#> [209] "katy"
#> [210] "katy"
#> [211] "katy"
#> [212] "katy"
#> [213] "katy"
#> [214] "katy"
#> [215] "katy"
#> [216] "katy"
#> [217] "katy"
#> [218] "katy"
#> [219] "katy"
#> [220] "katy"
#> [221] "katy"
#> [222] "katy"
#> [223] "katy"
#> [224] "katy"
#> [225] "katy"
#> [226] "katy"
#> [227] "katy"
#> [228] "katy"
#> [229] "katy"
#> [230] "katy"
#> [231] "katy"
#> [232] "katy"
#> [233] "katy"
#> [234] "katy"
#> [235] "katy"
#> [236] "katy"
#> [237] "katy"
#> [238] "katy"
#> [239] "katy"
#> [240] "katy"
#> [241] "katy"
#> [242] "katy"
#> [243] "katy"
#> [244] "katy"
#> [245] "katy"
#> [246] "katy"
#> [247] "katy"
#> [248] "katy"
#> [249] "katy"
#> [250] "katy"
#> [251] "katy"
#> [252] "katy"
#> [253] "katy"
#> [254] "katy"
#> [255] "katy"
#> [256] "katy"
#> [257] "katy"
#> [258] "katy"
#> [259] "katy"
#> [260] "katy"
#> [261] "katy"
#> [262] "katy"
#> [263] "katy"
#> [264] "katy"
#> [265] "katy"
#> [266] "katy"
#> [267] "katy"
#> [268] "katy"
#> [269] "katy"
#> [270] "katy"
#> [271] "katy"
#> [272] "katy"
#> [273] "katy"
#> [274] "katy"
#> [275] "katy"
#> [276] "katy"
#> [277] "katy"
#> [278] "katy"
#> [279] "katy"
#> [280] "katy"
#> [281] "katy"
#> [282] "katy"
#> [283] "katy"
#> [284] "katy"
#> [285] "katy"
#> [286] "katy"
#> [287] "katy"
#> [288] "katy"
#> [289] "katy"
#> [290] "katy"
#> [291] "katy"
#> [292] "katy"
#> [293] "katy"
#> [294] "katy"
#> [295] "katy"
#> [296] "katy"
#> [297] "katy"
#> [298] "katy"
#> [299] "katy"
#> [300] "katy"
#> [301] "katy"
#> [302] "katy"
#> [303] "katy"
#> [304] "katy"
#> [305] "katy"
#> [306] "katy"
#> [307] "katy"
#> [308] "katy"
#> [309] "katy"
#> [310] "katy"
#> [311] "katy"
#> [312] "katy"
#> [313] "katy"
#> [314] "katy"
#> [315] "katy"
#> [316] "katy"
#> [317] "katy"
#> [318] "katy"
#> [319] "katy"
#> [320] "katy"
#> [321] "katy"
#> [322] "katy"
#> [323] "katy"
#> [324] "katy"
#> [325] "katy"
#> [326] "katy"
#> [327] "katy"
#> [328] "katy"
#> [329] "katy"
#> [330] "katy"
#> [331] "katy"
#> [332] "katy"
#> [333] "katy"
#> [334] "katy"
#> [335] "katy"
#> [336] "katy"
#> [337] "katy"
#> [338] "katy"
#> [339] "katy"
#> [340] "katy"
#> [341] "katy"
#> [342] "katy"
#> [343] "katy"
#> [344] "katy"
#> [345] "katy"
#> [346] "katy"
#> [347] "katy"
#> [348] "katy"
#> [349] "katy"
#> [350] "katy"
```

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 confusion 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 comfusion 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],
       [ 5, 22]], 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(scorestree)
...: 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
...: #%% covering 95
...: #%% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean() ,
...: scores.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='sqrt', max_depth=None, min_samples_leaf=1,
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 (%.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 (+/- 0.06)
Max features: 1, num estimators: 10, accuracy: 0.88 (+/- 0.06)
Max features: 1, num estimators: 14, accuracy: 0.88 (+/- 0.12)
Max features: 1, num estimators: 18, accuracy: 0.88 (+/- 0.06)
Max features: 1, num estimators: 22, accuracy: 0.87 (+/- 0.07)
Max features: 3, num estimators: 26, accuracy: 0.88 (+/- 0.06)
Max features: 3, num estimators: 30, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 34, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 38, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 42, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 46, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 50, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 54, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 58, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 62, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 66, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 70, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 74, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 78, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 82, accuracy: 0.89 (+/- 0.03)
Max features: 3, num estimators: 86, accuracy: 0.89 (+/- 0.06)
Max features: 3, num estimators: 90, accuracy: 0.89 (+/- 0.03)

```

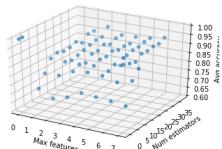
Gambar 4.139 Pengamatan Program

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:\VSemester\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(2, 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

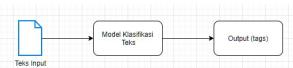
<https://youtu.be/Zsk8IXYg7Gk>

4.11 1174071 - Muhammad Abdul Gani Wijaya

4.11.1 Teori

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

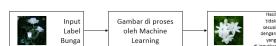
Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk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

Index	COMMENT_ID	AUTHOR	DATE	CONTENT	CLASS
0	r12hunedead.	Cory Wilson	2015-09-28T2...	re	0
1	r1249xmas4d.	Iain Gating	2015-09-28T2...	re	0
2	r125tccy5qjh...	Led Public	2015-09-28T2...	I'm trying to ha...	1
3	r12ftrwphpm...	Cheryl Fox	2015-09-28T2...	Rock...lol...	0
4	r12gcvk4ew...	PATICK_TW	2015-09-28T2...	Party rock	0
5	r12jrolymex...	Brian Bral	2015-09-28T0...	Shuffle	0
6	r12kflll0...	Brian Bral	2015-09-28T0...	Omg	0
7	r12l31obegy...	Alex Derev	2015-09-28T0...	Just really ..	0
8	r12mgw513l...	Slimmer	2015-09-28T0...	Assassinate />	0
9	r12llemcn0m...	Silvia Bauer	2015-09-28T0...	Incredible so...	0
10	r12ntryr0av...	Uniteide	2015-09-28T0...	I love you so much	0
11	r12og1ej5ut...	genti maria	2015-09-28T0...	2BIS L1CEEE	0
12	r12rtrxtet...	Alex Martin	2015-09-28T0...	I miss you so...	0
13	r12ubmztrv...	Alex Jansen	2015-09-28T0...	2BIS the...	0
14	r12uf4k2x0wy...	Aliden Hill	2015-09-27T2...	Best song of...	0
15	r12rt5qy15n...	joe paulin	2015-09-27T2...	super nice,	0
16	r12zg7jkxav...	bilal kemo	2015-09-27T2...	unbelievable	0
17	r13f7xng8ll...	SoulInTheDunes	2015-09-27T2...	PARTY ROCK	0
18	r13y5qipm...	Phuong Lin	2015-09-27T2...	Thums up !	0
19	r13zf7qipm...	Gontalo	2015-09-27T2...	THIS IS	0
20	r13zw5qipm...	katya70N	2015-09-27T2...	LOL!!!!!!	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: 150, accuracy: 0.43 (-/-: 0.02)
Max features: 35, num estimators: 70, accuracy: 0.44 (-/-: 0.04)
Max features: 35, num estimators: 100, accuracy: 0.44 (-/-: 0.04)
Max features: 35, num estimators: 130, accuracy: 0.46 (-/-: 0.04)
Max features: 35, num estimators: 150, accuracy: 0.46 (-/-: 0.04)
Max features: 35, num estimators: 170, accuracy: 0.46 (-/-: 0.04)
Max features: 35, num estimators: 190, 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 (-/-: 0.03)
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 (-/-: 0.04)
Max features: 40, num estimators: 150, accuracy: 0.45 (-/-: 0.04)
Max features: 40, num estimators: 170, accuracy: 0.45 (-/-: 0.04)
Max features: 40, num estimators: 190, accuracy: 0.45 (-/-: 0.03)
Max features: 40, num estimators: 200, accuracy: 0.45 (-/-: 0.03)
Max features: 45, num estimators: 10, accuracy: 0.34 (-/-: 0.02)
Max features: 45, num estimators: 30, accuracy: 0.43 (-/-: 0.03)
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 (-/-: 0.04)
Max features: 45, num estimators: 150, accuracy: 0.45 (-/-: 0.04)
Max features: 45, num estimators: 170, accuracy: 0.45 (-/-: 0.04)
Max features: 45, num estimators: 190, accuracy: 0.45 (-/-: 0.03)
Max features: 45, num estimators: 200, accuracy: 0.45 (-/-: 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

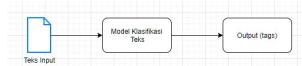
<https://youtu.be/X-xd9Nb78Gs>

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.

- 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 :



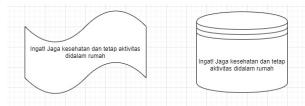
Gambar 4.170 Klasifikasi teks Youtube.

- Jelaskan apa yang dimaksud vektorisasi data.

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

- 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.

- 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.

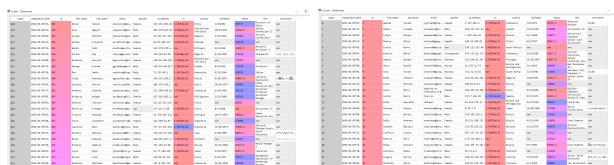
2. Soal 2

```

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	11261089pd.. Corey Wilson	2015-05-28T23..	ca...n't stop myself from...ing you	0	
1	11247442sd.. fatis Ganting	2015-05-28T23..	hey guys, i'm not...ed but	0	
2	11237254yjB.. Lek Music	2015-05-28T23..	hey guys, i'm not...ed but	1	
3	11237089pa.. Chevy Fox	2015-05-28T23..	Part 2, Rock...is cool...	0	
4	11236445ad.. PANTICO_7U	2015-05-28T23..	Party Rock	0	
5	11236144ew.. Brian Brad	2015-05-28T23..	Shoutin	0	
6	11236116oy.. Brian Brad	2015-05-28T23..	Shoutin	0	
7	11233100mg.. Alex Dafeo	2015-05-28T23..	This song is...t's been a while	0	
8	11230663tl.. GLOWARD	2015-05-28T23..	Just really...a	0	
9	11230600.. Jlement	2015-05-28T23..	January />	0	
10	11230000.. Silvia Basilio	2015-05-28T23..	Incredible no...s	0	
11	11227479nd.. Uniteinside	2015-05-28T23..	song so much	0	
12	11224454nt.. NoBigDeal	2015-05-28T23..	2015 Lifef	0	
13	11219327rv.. Alex Martin	2015-05-28T23..	people dress,	0	
14	11218127rv.. Alex Jensen	2015-05-28T23..	last year of,	0	
15	11216130ny.. Adam Hill	2015-05-27T22..	Best song	0	
16	11215701yj.. Joe pebb	2015-05-27T22..	super nice,	0	
17	11214742ev.. Silvia Kewo	2015-05-27T22..	super awesome	0	
18	11213701ll.. SoulInTheDunes	2015-05-27T21..	PARTY ROCK	0	
19	11213454nc.. Phuong Linh	2015-05-27T21..	Thumbs up if	0	
20	11212710pm.. Gonzalo	2015-05-27T21..	this is	0	
	11212640.. ktmr7QH	2015-05-27T21..	LMAO!!!!!!	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 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 :

Gambar 4.180 Hasil Soal 8

4.12.3 Penanganan Error

1. ScreenShoot Error

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

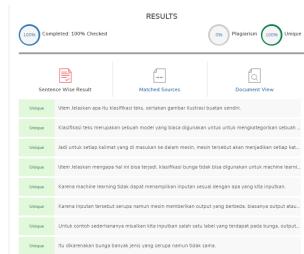
Gambar 4.181 SyntaxError

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

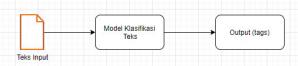
<https://youtu.be/WA8JwD2lzPw>

4.13 1174067 - Kaka Kamaludin

4.13.1 Teori

4.13.1.1 Klasifikasi Teks

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 4.183 Klasifikasi Teks.

4.13.1.2 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 adalah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut:



Gambar 4.184 Klasifikasi Bunga.

4.13.1.3 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:



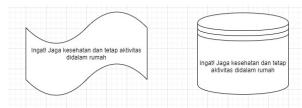
Gambar 4.185 Klasifikasi Teks pada Youtube.

4.13.1.4 Vektorisasi Data

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

4.13.1.5 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 4.186 Bag of Words.

4.13.1.6 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 4.187 TF-IDF.

4.13.2 Praktek

4.13.2.1 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 csv = pd.read_csv('D:/OneDrive - Hybi.god/KULIAH/Semester 6/AI/
    KB3C/src/1174067/4/csv.csv', sep=',') #membaca file csv
    dimana data pada file csv dipisahkan oleh koma, lalu
    ditampung di variable csv.

```

Listing 4.12 kodingan praktek no. 1

Index	registration_dttr	id	first_name	last_name	email
0	2016-02-03T0...	1	Amanda	Jordan	ajordan0@...
1	2016-02-03T1...	2	Albert	Freeman	afreeman1...
2	2016-02-03T0...	3	Evelyn	Morgan	emorgan2@...
3	2016-02-03T1...	4	Denise	Riley	driley3@...
4	2016-02-03T0...	5	Carlos	Burns	cburns4@...
5	2016-02-03T0...	6	Kathryn	White	kwhite5@...
6	2016-02-03T0...	7	Samuel	Holmes	sholmes6@...
7	2016-02-03T0...	8	Harry	Howell	hhowell7@...
8	2016-02-03T0...	9	Jose	Foster	jfoster8@...
9	2016-02-03T1...	10	Emily	Stewart	estewart9@...
10	2016-02-03T1...	11	Susan	Perkins	sperkins10@...
11	2016-02-03T1...	12	Alice	Berry	aberryb11@...
12	2016-02-03T1...	13	Justin	Berry	jberry12@...
13	2016-02-03T2...	14	Kathy	Reynolds	Unknown

Format Resize Background color Column min/max Save and Close Close

Gambar 4.188 hasil praktek soal no. 1

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

```
1 csv1, csv2 = csv [:450], csv [450:] #membagi data menjadi dua bagian, variabel csv1 untuk menampung 450 baris data pertama, variabel csv2 untuk menampung 50 baris data terakhir.
```

Listing 4.13 kodingan praktek no. 2

Name	Type	Size	Value
csv	DataFrame	(499, 13)	Column names: registration_dttr, id, first_name, last_name, email, gen...
csv1	DataFrame	(450, 13)	Column names: registration_dttr, id, first_name, last_name, email, gen...
csv2	DataFrame	(49, 13)	Column names: registration_dttr, id, first_name, last_name, email, gen...

Gambar 4.189 hasil praktek soal no. 2

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

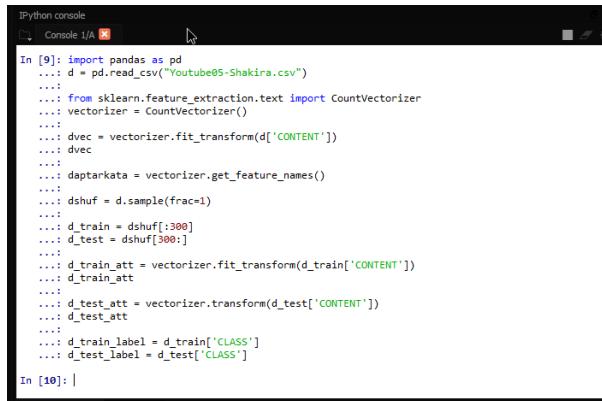
```
1 print(1174067%4) #hasilnya 3, Shakira
```

Listing 4.14 1174067 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 4.15 kodingan praktek no. 3


The screenshot shows an IPython console window with the title "IPython console". The console has tabs for "Console I/A" and "File". The code in the cell (In [9]) is identical to the one in Listing 4.15. The output of the cell shows the execution of each line of code, from importing pandas to defining d_train_label and d_test_label. The final line, "In [10]: |", indicates the cell is still active.

```

In [9]: 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' ]
In [10]: |

```

Gambar 4.190 hasil praktek soal no. 3(1)

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', '10171377578919894134' ...]
dshuf	DataFrame	(370, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS

Gambar 4.191 hasil praktek soal no. 3(2)

4.13.2.4 Cobalah klarifikasi dari data vektorisasi yang ditentukan di nomor sebelumnya dengan klasifikasi SVM. Tunjukkan keluarannya 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 4.16 kodingan praktek no. 4

```
In [10]: 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: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)
Out[10]: 0.6142857142857143
```

In [11]:

Gambar 4.192 hasil praktek soal no. 4

4.13.2.5 Cobalah klasifikasi 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 4.17 kodingan praktek no. 5

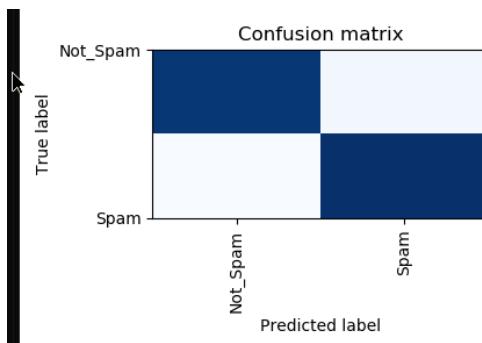
```
In [11]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
...: clftree.score(d_test_att, d_test_label)
Out[11]: 0.9142857142857143
```

Gambar 4.193 hasil praktek soal no. 5

4.13.2.6 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_labelstree = clftree.predict(d_test_att)
3 cmmtree = confusion_matrix(d_test_label, pred_labelstree)
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(cmmtree, classes=types, normalize=True)
46 plt.show()
```

Listing 4.18 kodingan praktek no. 6(1)



Gambar 4.194 hasil praktek soal no. 6(1)

Plot confusion matrix dari klasifikasi Decission Tree.

```

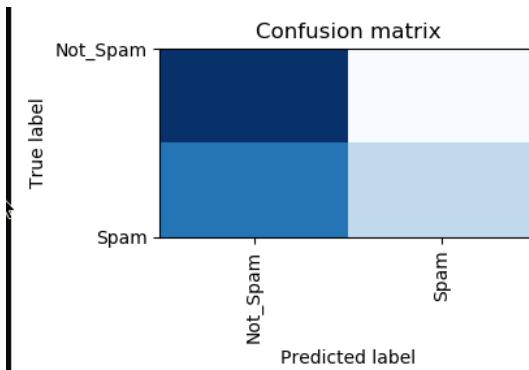
1 from sklearn.metrics import confusion_matrix
2 pred_labelssvm = clfsvm.predict(d_test_att)
3 cmsvm = confusion_matrix(d_test_label, pred_labelssvm)
4 cmsvm
5
6 import matplotlib.pyplot as plt
7 import itertools
8 def plot_confusion_matrix(cm, classes,
9                           normalize=False,
10                           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     #)

```

```

37
38     plt.tight_layout()
39     plt.ylabel('True label')
40     plt.xlabel('Predicted label')
41
42 types = pd.read_csv("classes.txt", sep='\s+', header=None, usecols=[1], names=['type'])
43 types = types['type']
44 types
45
46 import numpy as np
47 np.set_printoptions(precision=2)
48 plt.figure(figsize=(4,4), dpi=100)
49 plot_confusion_matrix(cmsvm, classes=types, normalize=True)
50 plt.show()

```

Listing 4.19 kodingan praktek no. 6(2)**Gambar 4.195** hasil praktek soal no. 6(2)

Plot confusion matrix dari klasifikasi SVM.

```

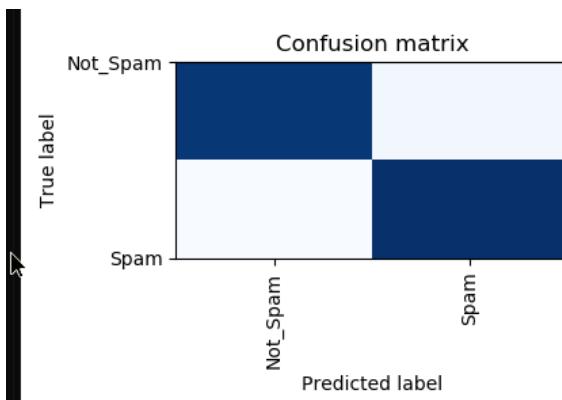
1 from sklearn.metrics import confusion_matrix
2 pred_labels = clf.predict(d_test_att)
3 cm = confusion_matrix(d_test_label, pred_labels)
4
5 import matplotlib.pyplot as plt
6 import itertools
7 def plot_confusion_matrix(cm, classes,
8                          normalize=False,
9                          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     #              horizontalalignment="center",
36     #              color="white" if cm[i, j] > thresh else "black"
37     #)
38
39     plt.tight_layout()
40     plt.ylabel('True label')
41     plt.xlabel('Predicted label')
42
43 types = pd.read_csv("classes.txt", sep='\s+', header=None, usecols
44     =[1], names=['type'])
45 types = types['type']
46 types
47
48 import numpy as np
49 np.set_printoptions(precision=2)
50 plt.figure(figsize=(4,4), dpi=100)
51 plot_confusion_matrix(cmtree, classes=types, normalize=True)
52 plt.show()

```

Listing 4.20 kodingan praktek no. 6(3)**Gambar 4.196** hasil praktek soal no. 6(3)

Plot confusion matrix dari klasifikasi Random Forest.

4.13.2.7 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)
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))

```

Listing 4.21 kodingan praktek no. 7

```

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

```

Gambar 4.197 hasil praktek soal no. 7

4.13.2.8 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(
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

```

```
14     print("Max features: %d, num estimators: %d, accuracy: %.2f (+/- %.2f)" % (max_features, n_estimators, scores.mean(), scores.std() * 2))
```

Listing 4.22 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 4.198 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: 0.93 (+/- 0.04)
Max features: 45, num estimators: 110, accuracy: 0.93 (+/- 0.05)
Max features: 45, num estimators: 130, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 150, accuracy: 0.93 (+/- 0.05)
Max features: 45, num estimators: 170, accuracy: 0.93 (+/- 0.04)
Max features: 45, num estimators: 190, accuracy: 0.93 (+/- 0.04)

```

Gambar 4.199 hasil praktek soal no. 8(2)

4.13.3 Penanganan Error

1. FileNotFoundError

```

file "spamdata_1100/powers.csv", line 489, in
spamdata_1100/powers.loadheader_setattr_powers_csv"
[Errno 2] [Errno 2] File b'\\Desktop\\spamdata_1100\\powers\\powers.csv' does not exist: b'\\Desktop\\spamdata_1100\\powers\\powers.csv'

```

Gambar 4.200 FileNotFoundError

2. Cara Penangan Error

- FileNotFoundError

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

4.13.4 Link Youtube

<https://www.youtube.com/playlist?list=PL4dhp4u89PHbhX9jrGyM3N12gmwhY3uIe>

4.14 1174054 - Aulyardha Anindita

4.14.1 Teori

1. Klasifikasi teks

Klasifikasi adalah suatu proses yang memiliki tugas terpenting dalam pemrosesan bahasa alami (natural language processing), pada proses ini berupa mengklasifikasikan string teks atau dokumen kedalam kategori yang berbeda yang tergantung pada konten string. dalam klasifikasi teks terdapat berbagai aplikasi seperti mendeteksi sentimen pengguna dari tweet, mengklasifikasikan email sebagai spam, mengklasifikasikan posting blog kedalam kategori yang berbeda, dan sebagainya. contoh dari klasifikasi teks seperti ketika kita akan mencari kata kucing, sedang, makan, ikan lalu jika kata yang dimaksud tersebut sesuai, maka akan menampilkan bilangan biner 1 dan jika salah akan menampilkan bilangan biner 0. seperti yang terlihat pada gambar berikut :



Gambar 4.201 Klasifikasi Teks

2. Mengapa Klasifikasi bunga tidak bisa menggunakan machine learning

Klasifikasi bunga tidak bisa menggunakan machine learning karena tidak semua bunga memiliki ciri-ciri yang sama, dalam kata lain terdapat data noise dalam klasifikasi bunga sehingga tidak bisa menggunakan machine learning. Contohnya seperti bunga terompet yang memiliki warna merah dengan jumlah kelopak bunga 5. lalu ada bunga warna merah lainnya dengan jumlah kelopak yang sama namun ternyata bukan bunga terompet disamping memiliki kategori yang banyak sekali. bahkan juga terdapat bunga yang tidak jelas dengan warna yang sesuai atau tidak, sehingga bisa menyebabkan data noise.



Gambar 4.202 Klasifikasi Bunga

3. Teknik pembelajaran mesin pada teks pada kata-kata yang digunakan di youtube

Kita ambil sebuah kasus yang semua orang telah ketahui dan juga pahami. Kasus tersebut yaitu perekondasian video dari pencarian menggunakan "text / kata" di Youtube. Pada saat menggunakan Youtube terdapat Machine Learning yang bekerja dan memproses perintah ataupun aktivitas tersebut, dimana akan memlter secara otomatis video yang disesuaikan dengan "keyword" yang kita masukkan sehingga memberikan keluaran video dengan keyword yang benar. Adapula tur yang di dapatkan ketika sedang menonton Youtube. Tampilan sebelah kanan terdapat pilihan 'Next' atapun 'Suggestion' yang menampilkan beberapa video serupa sesuai dengan yang dicari atau sedang ditonton. Ketika menglik salah satu video dari baris tersebut, maka Youtube akan mengingat dan menggunakan kata yang tertera sebagai referensi kembali sehingga akan memberikan kemudahan pada pencarian yang lainnya, Dan disitulah mesin belajar sendiri dan menyimpan data secara berkala sehingga berkembang.



Gambar 4.203 Teknik Pembelajaran Mesin

4. Vektorisasi Data

Vektorisasi data adalah salah satu proses pembagian dan pemecahan data yang kemudian dilakukan perhitungan. Vektorisasi dapat juga diartikan sebagai setiap data yang mungkin dipetakan ke integer tertentu dan jika memiliki array yang cukup besar maka setiap kata atau data yang cocok dengan slot unik dalam array.

5. Bag Of Words

Bag of words adalah sebuah gambaran yang sederhana yang digunakan untuk pengolahan bahasa alami dan juga pencarian suatu informasi, yang juga dikenal sebagai model ruang vektor, pada model ini tiap kalimat dalam dokumen digambarkan sebagai token kemudian mengabaikan tata bahasa bahkan urutan kata tapi mampu menghitung frekuensi kejadian

atau kemunculan kata dari dokumen.

	intelligent	application	creates	business	processes	bots	area	i	an	intelligence
Doc 1	2	1	1	1	0	0	0	0	0	0
Doc 2	0	0	0	0	0	0	0	1	0	0
Doc 3	0	0	0	1	0	0	0	1	1	1

Gambar 4.204 Bag Of Words

6. TF-IDF

TF-IDF merupakan proses atau tahap yang memberikan frekuensi kata dalam setiap dokumen dalam korpus atau mengganti data menjadi number. Hal ini berupa rasio yang berapa kali muncul dalam dokumen dibandingkan dengan jumlah total kata dalam dokumen. hal ini meningkat seiring jumlah kemunculan kata didalam dokumen juga meningkat. setiap dokumen memiliki tf sendiri. berikut contoh 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 4.205 TF-IDF

4.14.2 Praktek

1. Nomor 1

```
1 # Nomor 1
2 import pandas as pd #import package pandas, lalu dialiaskan
    menjadi pd.
3 sp = pd.read_csv('D:/Mata Kuliah/Tingkat 3/Semester 6/
    Kecerdasan Buatan/Chapter 4/datadummy.csv', delimiter = ',',
    ) #membaca file csv dimana data pada file csv dipisahkan
    oleh koma, lalu ditampung di variable sp.
```

Hasilnya :

Name	Type	Size	Value
sp	Dataframe	(500, 8)	Column names: gender, race/ethnicity, parental level of education, in ...

Gambar 4.206 Hasil Nomor 1

2. Nomor 2

```
1 # Nomor 2
2 sp1, sp2 = sp[:450], sp[450:] #membagi data menjadi dua
    bagian, variable sp1 untuk menampung 450 baris data
    pertama, variable sp2 untuk menampung 50 baris data
    terakhir.
```

Hasilnya :

Name	Type	Size	Value
sp	Dataframe	(500, 8)	Column names: gender, race/ethnicity, parental level of education, ...
sp1	Dataframe	(450, 8)	Column names: gender, race/ethnicity, parental level of education, ...
sp2	Dataframe	(50, 8)	Column names: gender, race/ethnicity, parental level of education, ...

Gambar 4.207 Hasil Nomor 2

3. Nomor 3

```

1 import pandas as pd
2 d = pd.read_csv("D:/Mata Kuliah/Tingkat 3/Semester 6/
    Kecerdasan Buatan/Chapter 4/Youtube04-Eminem.csv")
3
4 from sklearn.feature_extraction.text import CountVectorizer
5 vectorizer = CountVectorizer()
6
7 dvec = vectorizer.fit_transform(d[ 'CONTENT'])
8 dvec
9
10 daptarkata = vectorizer.get_feature_names()
11
12 dshuf = d.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']
d_test_label = d_test[ 'CLASS']
22
23
24

```

Hasilnya :

Name	Type	Size	Value
d	Dataframe	(440, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
d_test	Dataframe	(100, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
d_train_label	Series	(340,)	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	1000	'sep', 'www', 'me', '12', '100', '1000', '10007700245414', ...
dshuf	Dataframe	(440, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
sp	Dataframe	(500, 8)	Column names: gender, race/ethnicity, parental level of education, ...
sp1	Dataframe	(450, 8)	Column names: gender, race/ethnicity, parental level of education, ...
sp2	Dataframe	(50, 8)	Column names: gender, race/ethnicity, parental level of education, ...

Gambar 4.208 Hasil Nomor 3

4. Nomor 4

```

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)

```

Hasilnya :

```

In [39]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
C:\Users\Hasee\PycharmProjects\Klasifikasi\venv\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
accommodate scaling for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
  "avoid this warning.", FutureWarning)
Out[39]: 0.8548048484848484

```

Gambar 4.209 Hasil Nomor 4

5. Nomor 5

```

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)

```

Hasilnya :

```

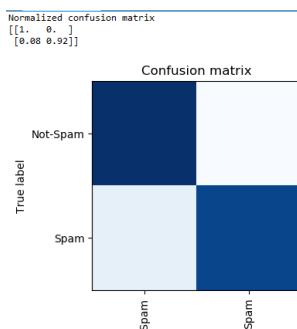
In [40]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
...: clftree.score(d_test_att, d_test_label)
Out[40]: 0.9527827827827827

```

Gambar 4.210 Hasil Nomor 5

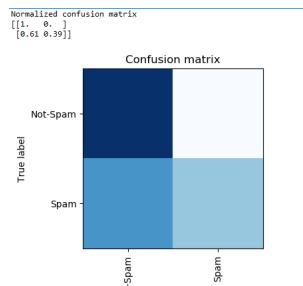
6. Nomor 6

Plot Confusion Matrix dari Decision Tree



Gambar 4.211 Hasil Nomor 6-1

Plot Confusion Matrix dari SVM



Gambar 4.212 Hasil Nomor 6-2

7. Nomor 7

```

1 np.set_printoptions(precision=2)
2 plt.figure(figsize=(4,4), dpi=100)
3 plot_confusion_matrix(cmsvm, classes=types, normalize=True)
4 plt.show()
5
6 # In []:
7 # Confusion Matrix - Random Forest
8 from sklearn.metrics import confusion_matrix
9 pred_labels = clf.predict(d_test_att)
10 cm = confusion_matrix(d_test_label, pred_labels)

```

Hasilnya :

```

Accuracy: 0.88 (<-- 0.85)
Accuracy: 0.87 (<-- 0.82)
C:\Users\AULYARDHA\OneDrive\Belajar\Kuliah\Python\sklearn\metrics\confusion_matrix.py:196: 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. (see https://scikit-learn.org/stable/modules/metrics.html#gamma)
"avoid this warning.", FutureWarning)
Accuracy: 0.86 (<-- 0.84)

```

Gambar 4.213 Hasil Nomor 7

8. Nomor 8

```

1 def plot_confusion_matrix(cm, classes,
2                           normalize=False,
3                           title='Confusion matrix',
4                           cmap=plt.cm.Blues):
5
6     """
7         This function prints and plots the confusion matrix.
8         Normalization can be applied by setting 'normalize=True'.
9     """
10    if normalize:
11        cm = cm.astype('float') / cm.sum(axis=1)[:, np.
12        newaxis]
13        print("Normalized confusion matrix")
14    else:
15        print('Confusion matrix, without normalization')

```

Hasilnya :

```
Max Features: 5, num estimators: 10, accuracy: 0.90 (+/- 0.02)
Max Features: 5, num estimators: 30, accuracy: 0.94 (+/- 0.02)
Max Features: 5, num estimators: 50, accuracy: 0.93 (+/- 0.02)
Max Features: 5, num estimators: 70, accuracy: 0.94 (+/- 0.03)
Max Features: 5, num estimators: 90, accuracy: 0.94 (+/- 0.03)
Max Features: 5, num estimators: 110, accuracy: 0.96 (+/- 0.02)
Max Features: 5, num estimators: 130, accuracy: 0.96 (+/- 0.02)
Max Features: 5, num estimators: 150, accuracy: 0.94 (+/- 0.02)
Max Features: 5, num estimators: 159, accuracy: 0.94 (+/- 0.02)
Max Features: 5, num estimators: 170, accuracy: 0.94 (+/- 0.02)
Max Features: 5, num estimators: 180, accuracy: 0.95 (+/- 0.02)
Max Features: 10, num estimators: 10, accuracy: 0.91 (+/- 0.05)
Max Features: 10, num estimators: 30, accuracy: 0.95 (+/- 0.03)
Max Features: 10, num estimators: 50, accuracy: 0.95 (+/- 0.03)
Max Features: 10, num estimators: 70, accuracy: 0.95 (+/- 0.04)
Max Features: 10, num estimators: 90, accuracy: 0.95 (+/- 0.03)
Max Features: 10, num estimators: 110, accuracy: 0.95 (+/- 0.03)
Max Features: 10, num estimators: 130, accuracy: 0.95 (+/- 0.04)
Max Features: 10, num estimators: 150, accuracy: 0.95 (+/- 0.04)
Max Features: 10, num estimators: 170, accuracy: 0.95 (+/- 0.04)
Max Features: 10, num estimators: 180, accuracy: 0.95 (+/- 0.04)
Max Features: 15, num estimators: 10, accuracy: 0.92 (+/- 0.05)
Max Features: 15, num estimators: 30, accuracy: 0.94 (+/- 0.05)
Max Features: 15, num estimators: 50, accuracy: 0.95 (+/- 0.03)
Max Features: 15, num estimators: 70, accuracy: 0.95 (+/- 0.02)
Max Features: 15, num estimators: 90, accuracy: 0.95 (+/- 0.02)
Max Features: 15, num estimators: 110, accuracy: 0.95 (+/- 0.04)
Max Features: 15, num estimators: 130, accuracy: 0.95 (+/- 0.04)
Max Features: 15, num estimators: 150, accuracy: 0.95 (+/- 0.04)
Max Features: 15, num estimators: 170, accuracy: 0.95 (+/- 0.04)
```

Gambar 4.214 Hasil Nomor 8

4.14.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundException: File 'D:/Data Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/Chapter4/datalumpy.csv' does not exist
```

Gambar 4.215 File Not Found Error

2. Tuliskan Kode Error dan Jenis Error

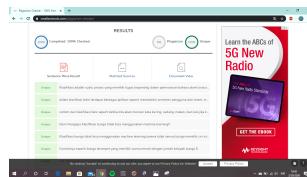
- File Not Found Error

3. Cara Penanganan Error

- File Not Found Error

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

4.14.4 Bukti Tidak Plagiat



Gambar 4.216 Bukti Plagiarisme

4.14.5 Link Youtube

<https://youtu.be/H7qxw-s1R0c>

4.15 Nurul Izza Hamka - 1174062

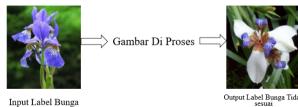
4.15.1 Teori

- Jelaskan apa itu klasifikasi teks, berserta gambar ilustrasi buatan sendiri
Klasifikasi teks adalah sebuah proses dalam penentuan kategori suatu dokumen teks disertai dengan karakteristik teks itu sendiri.



Gambar 4.217 Gambar Klasifikasi Data

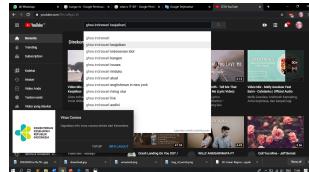
- Jelaskan mengapa klasifikasi bunga tidak bisa menggunakan machine learning, sertakan ilustrasi sendiri.
Karna Machine Learning tidak dapat menampilkan atau tidak dapat memberikan output sesuai dengan apa yang telah inputkan sebelumnya. hal ini terjadi karena mesin yang membeikan output atau yang biasa dikenal dengan noise. Contohnya adalah Ketika kita menginputkan sebuah label yang di dalamnya berupa Bungan, akan tetapi nantinya output yang di proses oleh mesin tersebut adalah labeh yang berbeda.



Gambar 4.218 Gambar Proses Klasifikasi Bunga

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

Salah satu Teknik pembelajaran yang ada di youtube adalah dengan menggunakan keyword. Seperti halnya Ketika kita mau mencari sebuah video pada youtube, kita tinggal mengetikkan sebuah keyword maka selanjutnya akan di proses untuk menampilkan hasil sesuai dengan keyword yang kita inputkan tadi.



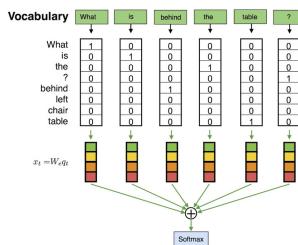
Gambar 4.219 Gambar Machine Learning Di Youtube

4. Jelaskan apa itu vektorisasi data.

Vektorisasi adalah sebuah proses konversi data raster menjadi sebuah vector yang umum, ini disebut dengan digitalisasi sedangkan aktifitasnya ini disebut dengan istilah digitasi. Vektorisasi adalah proses mengubah algoritma dari operasi pada nilai tunggal pada suatu waktu untuk beroperasi pada satu set nilai pada satu waktu.

5. Jelaskan apa itu Bag of Words dengan kata-kata sederhana dan ilustrasi sendiri.

BOW adalah singkatan dari Bag of Words yang merupakan sebuah metode untuk mengekstrak sebuah fitur dari dokumen teks. Fitur ini dapat kita gunakan dalam pelatihan machine learning algorithm. Fitur yang menciptakan kosakata dari semua kata yang unik disemua dokumen.



Gambar 4.220 Gambar Rumus Bag of Words

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

TF-IDF adalah singkatan dari Term Frequency - Inverse Term Frequency. Bagian TF menghitung berapa kali suatu kata terjadi dalam korpus yang diberikan. Karena corpus terdiri dari banyak dokumen, setiap dokumen dan kata-katanya akan memiliki jumlah TF mereka sendiri. Bagian IDF menghitung seberapa jarang suatu kata muncul di dalam dokumen.

$$w_{i,j} = t f_{i,j} \times \log \left(\frac{N}{d f_i} \right)$$

$t f_{i,j}$ = number of occurrences of i in j
 $d f_i$ = number of documents containing i
 N = total number of documents

Gambar 4.221 Gambar TF-IDF

4.15.2 Praktek Program

- Aplikasi sederhana menggunakan pandas, membuat data dummy format csv sebanyak 500 baris dan melakukan load dataframe pandas.

```

1 %% Soal 1
2
3 import pandas as pd #digunakan untuk mengimport library
4 pd = pd.read_csv("D:/LEC/IT SMT VI/ARTIFICIAL INTELLIGENCE/
Chapter 4/src/csv_izza.csv") #membaca file csv

```

Gambar 4.222 Gambar Hasil Soal 1

2. Dari data frame No 1 dipecah menjadi dua dataframe yaitu 450 row pertama dan 50 row sisanya.

```

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

```

Gambar 4.223 Gambar Hasil 450 Row Pertama

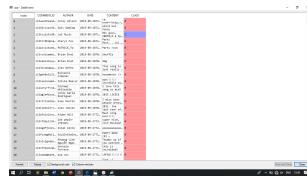
Gambar 4.224 Gambar Hasil 50 Row Sisanya

3. Vektorisasi dan klasifikasi data dengan Desicion Tree.

```

1 %% Soal 3
2
3 import pandas as izza #untuk import library pandas berguna
4         untuk mengelola dataframe
4 izza = izza.read_csv("D:/LEC/IT SMT VI/ARTIFICIAL
5 INTELLIGENCE/Chapter 4/src/Youtube03-LMFAO.csv") #membaca
6         file dengan format csv
7
8
9 from sklearn.feature_extraction.text import CountVectorizer #
10        untuk import countvectorizer berfungsi untuk memecah data
11        tersebut menjadi sebuah kata yang lebih sederhana
12 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
13        tersebut , pada code ini tidak ada hasilnya dikarenakan
14        spyder tidak mendukung hasil dari instasiasi.
15
16 dvec = vectorizer.fit_transform(izza[ 'CONTENT' ]) #untuk
17         melakukan pemecahan data pada dataframe yang terdapat pada
18         kolom konten
19 dvec #Untuk menampilkan hasil dari code sebelumnya
20
21 Daptarkata= vectorizer.get_feature_names()
22
23 dshuf = izza.sample(frac=1)
24
25 d_train=dshuf[:300]
26 d_test=dshuf[300:]
27
28 d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT' ])
29 d_train_att
30
31 d_train_label=d_train[ 'CLASS' ]
32 d_test_label=d_test[ 'CLASS' ]

```



Gambar 4.225 Gambar Hasil Soal 3

4. Mengklasifikasikan data vektorisasi dengan klasifikasi SVM.

```

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

```

```

In [24]: from sklearn import svm
...: clfsvm = svm.SVR(gamma = 'auto')
...: clfsvm.fit(d_train_att , d_train_label)
Out[24]:
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 4.226 Gambar Hasil Soal 4

5. Mengklasifikasi data vektorisasi dengan klasifikasi Desicion tree.

```

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

```

```

In [25]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att , d_train_label)
Out[25]:
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_samples_split=2,
min_weight_fraction_leaf=0.0, presort=False,
random_state=None, splitter='best')

```

Gambar 4.227 Gambar Hasil Soal 5

6. Plot confusion Matrix Menggunakan matplotlib.

```

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
19         print("Normalized confusion matrix")

```

```

19     else :
20         print('Confusion matrix , without normalization')
21
22     print(cm)

```

```

In [27]: 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')
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)

```

Gambar 4.228 Gambar Hasil Soal 6

7. Jalankan cross validation

```

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=5)
6
7 skor_rata2=scores.mean()
8 skoresd=scores.std()

```

```

In [28]: 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 4.229 Gambar Hasil Soal Nomor 7

8. Membuat program pengamatan komponen informasi .

```

1 #%%soal 8
2
3 max_features_opts = range(5, 50, 5) #max_features_opts sebagai variabel untuk membuat range 5,50,5
4 n_estimators_opts = range(10, 200, 20) #n_estimators_opts sebagai variabel untuk membuat range 10,200,20
5 rf_params = izza.empty((len(max_features_opts)*len(
6     n_estimators_opts),4), float) #rf_params sebagai variabel 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         clftrree = RandomForestClassifier(max_features=
11             max_features, n_estimators=n_estimators) #menampilkan variabel csf

```

```

10         scores = cross_val_score(clf , df_train_att ,
11             df_train_label , cv=5) #scores sebagai variabel training
12             rf_params[i,0] = max_features #index 0
13             rf_params[i,1] = n_estimators #index 1
14             rf_params[i,2] = scores.mean() #index 2
15             rf_params[i,3] = scores.std() * 2 #index 3
16             i += 1 #dengan ketentuan i += 1
17             print("Max features: %d, num estimators: %d, accuracy
: %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
mean(), scores.std() * 2))
# print hasil pengulangan yang sudah ditentukan

```

4.15.3 Penanganan Error

1. Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat dibaca atau dipanggil file csvnya.

FileNotFoundError: [Errno 2] File b'F:/Semester 6/Artificial Intelligence/Tags/4/scr/fancy.csv' does not exist: b'F:/Semester 6/Artificial Intelligence/Tags/4/scr/fancy.csv'

Gambar 4.230 Gambar Error

4.15.4 Bukti Tidak Plagiat

1. Berikut adalah gambar bukti tidak melakukan Plagiat



Gambar 4.231 Gambar Bukti Tidak Plagiat

4.15.5 Link Youtube

1. Berikut adalah lampiran Link Youtube untuk Chapter 4
<https://www.youtube.com/watch?v=zX3W99EB2ks>

BAB 5

CHAPTER 5

5.1 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 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



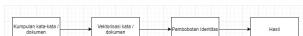
Gambar 5.2 Kecerdasan Buatan.

5.2 1174066 - D.Irga B. Naufal Fakhri

5.2.1 Teori

5.2.1.1 Jelaskan kenapa kata-kata harus di lakukan vektorisasi

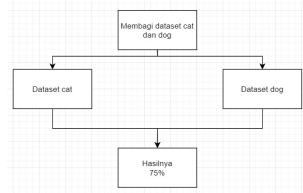
Kata kata harus dilakukan vektorisasi dikarenakan atau bertujuan utk mengukur nilai kemunculan suatau kata yang serupa dari sebuah kalimat sehingga kata-kata tersebut dapat di prediksi kemunculannya. atau juga di buatnya vektorisasi data digunakan untuk memprediksi bobot dari suatu kata misalkan ayam dan kucing sama-sama hewan maka akan dibuat prediksi apakah kata tersebut akan muncul pada kalimat yang kira-kira memiliki bobot yang sama.



Gambar 5.3 Teori 1

5.2.1.2 Jelaskan mengapa dimensi dari vektor dataset google bisa sampai 300.

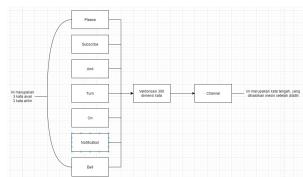
Karena setiap objek memiliki identitas khusus, misalnya sederhana. Dalam dataset Google ini memiliki 3 objek, kucing, anjing, dan ember yang disetuju. Kemudian dari masing-masing objek dibandingkan dataset antara kucing dan anjing kemudian kucing dan ember. Hasil yang diperoleh untuk kucing dan anjing sekitar 75% sedangkan untuk kucing dan ember yang memiliki persentase 15%. Untuk lebih lengkapnya bisa dilihat pada gambar berikut:



Gambar 5.4 Teori 2

5.2.1.3 Jelaskan konsep vektorisasi untuk kata

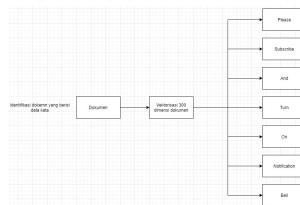
Konsep untuk vektorisasi kata sama dengan masukan atau input pada kata-kata di mesin pencari. Maka anggaplah itu akan dikeluarkan sebagai saran tentang kata tersebut. Jadi kata data diperoleh dari hasil yang diolah dalam kalimat sebelumnya yang telah diproses. Contoh sederhana dalam kalimat berikut (Please Subscribe and Turn on Notification Bell), dalam kalimat itu terkait dengan kalimat saluran, kata akan dibuat data pelatihan untuk mesin. Jadi kita kompilasi kata channel, maka mesin akan menampilkan hubungannya dengan kata tersebut. Contoh ilustrasi sederhananya seperti berikut:



Gambar 5.5 Teori 3

5.2.1.4 Jelaskan konsep vektorisasi untuk dokumen

Sama halnya dengan vektorisasi kata, yang membedakan hanya pada proses awalnya. Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, lalu kalimat yang terdapat pada dokumen akan dipecah menjadi kata-kata. Perhatikan gambar berikut:



Gambar 5.6 Teori 4

5.2.1.5 Jelaskan apa mean dan standar deviasi

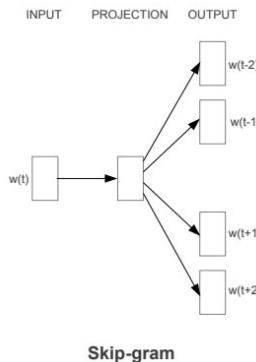
Mean adalah nilai rata-rata. Untuk mendapatkan makna ini, kita hanya perlu menambahkan data yang tersedia dan kemudian membaginya dengan jumlah data. Sedangkan standar deviasi adalah nilai statistik yang digunakan untuk menentukan bagaimana data didistribusikan dalam sampel, dan menentukan titik data individu dengan nilai rata-rata nilai sampel. Rumusnya ialah sebagai berikut:

Mean $\bar{X} = \frac{x_1 + x_2 + \dots + x_n}{n}$ atau $\bar{X} = \frac{\sum_{i=1}^n X_i}{n}$	\bar{X} = rata-rata $\sum_{i=1}^n X_i$ = jumlah seluruh nilai data n = jumlah seluruh frekuensi
Rumus Varian	
$s^2 = \frac{n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2}{n(n-1)}$	
Rumus Standar Deviasi	
$s = \sqrt{\frac{n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2}{n(n-1)}}$	

Gambar 5.7 Teori 5

5.2.1.6 Jelaskan apa itu skip-gram

Skip-gram sama halnya dengan vektorisasi kata, namun untuk skip-gram ia dibalik prosesnya. Yang sebelumnya dari kalimat lalu diolah untuk menemukan salah satu kata, kali ini dari keyword tersebut akan diolah menjadi suatu kalimat yang memiliki keterkaitannya dengan keyword tersebut. Contoh sederhananya seperti gambar berikut:



Gambar 5.8 Teori 6

5.2.2 Praktek

5.2.2.1 Nomor 1

```

1 import gensim, logging
2 logging.basicConfig(format='%(asctime)s : %(levelname)s : %(
     message)s', level=logging.INFO)
3 model_dirga = gensim.models.KeyedVectors.load_word2vec_format('N
     :/KB/GoogleNews-vectors-negative300.bin', binary=True, limit
     =500000)
  
```

Kode digunakan untuk import library gensim. Gensim itu sendiri berguna untuk melakukan pemodelan dengan dataset atau topik yang telah ditentukan. Untuk keluaran 100 logging itu library opsional karena logging hanya untuk menampilkan berupa log untuk setiap code yang dijalankan. Dan keluaran 101 itu hasil load data dari file vektor google itu, disini saya menggunakan limit karena kondisi laptop yang tidak mempunyai untuk melakukan running data sebesar 3 juta file, jadi saya batasi hanya melakukan running 500 ribu data saja, hasilnya ialah sebagai berikut:

```

[In 1]: import gensim, logging
        ....
        ....
        ....
[In 1]: model_dirga = gensim.models.KeyedVectors.load_word2vec_format('N
         :/KB/GoogleNews-vectors-negative300.bin', binary=True, limit=500000)
[2020-04-09 20:21:09,387 : DEBG : Loading projection weights from N
         :/KB/GoogleNews-vectors-negative300.bin
[2020-04-09 20:21:09,387 : DEBG : Index (500000, 300) matrix from N
         :/KB/GoogleNews-vectors-negative300.bin
  
```

Gambar 5.9 Hasil dari Nomor 1-1

```

1 model_dirga['love']
2 #%%
3 model_dirga['faith']
4 #%%
5 model_dirga['fall']
6 #%%
  
```

```

7 model_dirga[ 'sick' ]
8 #%%
9 model_dirga[ 'clear' ]
10 #%%
11 model_dirga[ 'shine' ]
12 #%%
13 model_dirga[ 'bag' ]
14 #%%
15 model_dirga[ 'car' ]
16 #%%
17 model_dirga[ 'wash' ]
18 #%%
19 model_dirga[ 'motor' ]
20 #%%
21 model_dirga[ 'cycle' ]

```

Menampilkan data hasil vektorisasi data:

```

In [4]: model_dirga['love']
Out[4]:
array([-0.18987274, -0.15234175,  0.02578791,  0.15939869,
       -0.15939869,  0.15939869,  0.15939869,  0.15939869,
       -0.02524512, -0.08203125, -0.02770906, -0.04304531, -0.23351555,
       0.16922188,  0.12098625,  0.15722855,  0.08756835, -0.08982422,
       -0.08982422,  0.12098625,  0.15722855,  0.08756835,  0.08756835,
       -0.05151625,  0.05151778,  0.10893359,  0.11161461,  0.16388594,
       -0.17146451,  0.13954444,  0.08625278,  0.08214649,  0.13954444,
       0.07744649,  0.13954444,  0.08625278,  0.08214649,  0.13954444,
       0.18652344, -0.28949964,  0.07808078,  0.02680989, -0.18645311,
       -0.18645311,  0.28949964,  0.07808078,  0.02680989,  0.08944961,
       -0.18986132,  0.14941486, -0.18693359,  0.01556396,  0.08944375,
       0.11230469, -0.04701117, -0.1176953,  -0.007784,  -0.01818848,
       0.01818848,  0.04701117,  0.1176953,  -0.007784,  -0.01818848,
       -0.09667369, -0.21972656, -0.00320864, -0.01918242,  0.18457031,
       0.09515255, -0.08539779, -0.11121601,  0.08361625, -0.36664862,
       -0.08958552,  0.08958552,  0.08958552,  0.08958552,  0.08958552,
       -0.11854688,  0.22558594,  0.08740234, -0.2265625, -0.29492188,
       0.08958552,  0.08958552,  0.08958552,  0.08958552,  0.08958552,
       -0.08958552,  0.23532812, -0.17871094, -0.12451172, -0.17285155,
       -0.11767578,  0.19729552, -0.03466797, -0.10489351, -0.164825,
       -0.164825,  0.19729552, -0.03466797, -0.10489351, -0.164825,
       0.08973524, -0.04828323,  0.03863965,  0.01623535, -0.1648252,
       -0.22167969,  0.171875,  0.12811719,  0.01935353,  0.4451125,
       0.4451125,  0.171875,  0.12811719,  0.01935353,  0.4451125,
       0.05566406, -0.05698883, -0.01422119,  0.15917969,  0.07421875,
       -0.148625, -0.13118594, -0.12792969,  0.12808547,  0.09583789,
       -0.08655695,  0.05761719, -0.08474266,  0.16992188,  0.13671875,
       -0.08662231,  0.08662231,  0.08662231,  0.08662231,  0.08662231,
       0.08662231, -0.01928711,  0.09423828, -0.13135954, -0.27929688,
       0.27734375,  0.31835958,  0.10858594,  0.11425781, -0.27734375,

```

Gambar 5.10 Data love

```

In [8]: model_dirga['faith']
Out[8]:
array([ 0.1051188, -0.01510391,  0.105115,  0.13476162, -0.14664536,
       0.12329251, -0.08345781,  0.10511562,  0.12207931,  0.12476562,
       0.06646825,  0.18945312, -0.16981562,  0.21679886, -0.27148458,
       0.3298125,  0.18448219,  0.36132812, -0.1953125, -0.18164602,
       0.18164602,  0.18448219,  0.36132812, -0.1953125, -0.18164602,
       -0.05957031, -0.22949219, -0.00804248,  0.26171875,  0.1892734,
       -0.1528125,  0.21484575,  0.0115524,  0.02111816,  0.18554688,
       0.18554688,  0.21484575,  0.0115524,  0.02111816,  0.18554688,
       0.3125,  0.06646025, -0.17675781, -0.01780864, -0.164625,
       0.08662231,  0.08662231,  0.08662231,  0.08662231,  0.08662231,
       -0.02524504,  0.16308594, -0.1183594, -0.08007312,  0.13085938,
       -0.02524504,  0.16308594, -0.1183594, -0.08007312,  0.13085938,
       0.2753062, -0.26605469, -0.10915662, -0.29214464, -0.1874 ,
       0.02331543,  0.02331543,  0.02331543,  0.02331543,  0.02331543,
       -0.11132812,  0.05688477,  0.12355899,  0.09144553,  0.04150891,
       -0.02331543,  0.02331543,  0.02331543,  0.02331543,  0.02331543,
       -0.20880781,  0.04199219,  0.05348661, -0.27734735,  0.09130859,
       -0.17080781,  0.04199219,  0.05348661, -0.19824219, -0.0883951,
       0.05348661, -0.07234257,  0.05348661, -0.19824219, -0.0883951,
       -0.13574219, -0.08078125,  0.0649971,  0.0666559, -0.29296875,
       -0.0666559,  0.0649971,  0.0666559, -0.29296875,
       -0.20695469, -0.09802331,  0.0534660 ,  0.05349461, -0.03222056,
       -0.29296875,  0.2558939,  0.0039298,  0.14625,  0.05189457,
       -0.05189457,  0.2558939,  0.0039298,  0.14625,  0.05189457,
       0.04644918,  0.04644918,  0.04644918,  0.04644918,  0.04644918,
       0.04644918,  0.04644918,  0.04644918,  0.04644918,  0.04644918,
       -0.05688477,  0.18409847, -0.04785156, -0.15136719, -0.07714844,
       -0.07714844,  0.18409847, -0.04785156, -0.15136719, -0.07714844,
       -0.03997878,  0.07373847, -0.01220795,  0.22466938,  0.14550781,
       -0.01220795,  0.22466938,  0.14550781,  0.03997878,
       -0.09521484,  0.07881328,  0.04783528,  0.375,  0.0234375,
       0.2265625, -0.12666547, -0.04296875,  0.328125,  0.15332081,

```

Gambar 5.11 Data faith

```
In [6]: model_dirge['fall']
Out[6]:
array([-0.04772461,  0.19742188, -0.09977344,  0.16904531, -0.1226152 ,
       -0.18693359,  0.84521289,  0.01904297,  0.14648438,  0.15830862,
      -0.08691486,  0.84492188,  0.015874,  0.86651406, -0.15824219,
      -0.18693359,  0.13971875,  0.015874,  0.86651406, -0.15824219,
     -0.1815625 ,  0.13971875,  0.09228516, -0.12109375,  0.12895312,
     0.03417969,  0.21869373,  0.01793593,  0.125 ,  0.05451489,
     0.04419921,  0.29828212, -0.18554688,  0.08486894, -0.02087482,
     0.12861719,  0.15136719,  0.0445557,  0.04463904,  0.27539894,
     -0.10539378,  0.05468774,  0.04958953,  0.05737375, -0.03258989,
     0.18295978, -0.15136719, -0.0044557,  0.04463904,  0.27539892,
     -0.0255227 ,  0.06279828,  0.07080878, -0.07617183,  0.05642969,
     0.01672363,  0.84711514,  0.19628986, -0.08984375,  0.078125 ,
     0.05424869,  0.8313 ,  0.12988211,  0.03279586,  0.14549781,
     0.08448889,  0.14108159,  0.04514816, -0.078125 ,  0.06988589,
     0.26757912,  0.82091953, -0.12895312, -0.04482812,  0.18945312,
     0.12861719,  0.15136719,  0.0445557,  0.04463904,  0.27539892,
     -0.14941486, -0.02331543, -0.03958978, -0.10480391, -0.14168156,
     -0.12861719,  0.83876372,  0.04598844, -0.209625 ,  0.03540809,
     0.12861719,  0.15136719,  0.0445557,  0.04463904,  0.27539892,
     0.0429875 ,  0.89932381,  0.08056641, -0.06268382,  0.12255895,
     0.08089711,  0.08482178, -0.1648625 , -0.03274484,  0.0701125 ,
     0.07958984, -0.12690625,  0.01879883, -0.17773434,  0.061203945,
     -0.0255227 ,  0.06279828,  0.07080878, -0.07617183,  0.05642969,
     -0.14453125,  0.10858853, -0.0853735 ,  0.18205975, -0.03938725,
     0.04853394,  0.80135514, -0.15828312, -0.09130859, -0.12255895,
     -0.04297759,  0.07470782,  0.11797579,  0.12898662,  0.04588944,
```

Gambar 5.12 Data fall

```
In [7]: model_dirge['sick']
Out[7]:
array([-0.82171886e-01,  1.49414862e-01, -4.05273439e-02,  1.64862500e-01,
       -5.5976525e-01,  3.2238525e-01, -1.73828125e-01, -1.47469518e-01,
       -0.12238662e-01,  1.2238662e-01,  0.12238662e-01,  0.12238662e-01,
       2.24304199e-01,  9.6679875e-02, -1.00005635e-01, -1.12384666e-01,
       1.66815625e-01, -1.79687908e-01,  5.92648165e-03, -2.45171888e-01,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       1.7871935e-01, -9.9182189e-01,  8.8671375e-02, -1.95312500e-01,
       1.2238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       9.42328312e-02, -3.12598000e-02,  1.98974699e-02, -6.39648415e-02,
       1.1685244e-01,  1.2394675e-01, -6.03627344e-02,  4.68754000e-01,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       5.79513281e-02, -1.04988469e-01,  1.69854512e-01, -0.08073125e-02,
       -0.01717875e-01,  1.06208152e-01,  0.12238662e-01,  0.12238662e-01,
       -0.01717875e-01,  0.05312528e-01,  5.01523438e-02, -2.018171588e-01,
       7.1299625e-02,  4.37911719e-02,  2.0597825e-01,  5.71239802e-02,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       2.4884675e-01, -6.5429875e-02, -2.02836712e-02,  1.52534770e-01,
       -0.57421775e-01,  0.82124037e-01, -2.00087712e-01, -5.05717004e-02,
       2.02836712e-02, -1.00087712e-01,  1.77734770e-01,
       -0.49414602e-02,  3.67769531e-02,  1.91692508e-01,  2.77734770e-01,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       4.66380594e-02, -5.1778125e-02,  1.63989538e-01,  4.17408459e-02,
       0.11717875e-02, -0.01717875e-01, -1.50755052e-01,  2.61713700e-01,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       4.19921875e-01, -6.88479562e-02,  9.42328312e-02, -1.96239062e-01,
       -0.12238662e-01,  1.53128312e-01,  1.318339538e-01, -1.81456250e-01,
       -0.61132812e-01, -1.53128312e-01,  1.318339538e-01, -1.81456250e-01,
       -0.12238662e-01, -1.43554688e-01,  8.30878125e-03, -9.81453112e-02,
       3.5599375e-01, -1.54312508e-01,  3.45915012e-02, -1.00087712e-01,
       1.15036484e-02, -2.77734770e-01,  2.60973254e-02,  0.027343751e-01,
```

Gambar 5.13 Data sick

```
In [8]: model_dirge['clear']
Out[8]:
array([-0.44148625e-04, -1.40496931e-01, -1.49414862e-01, -4.24846458e-02,
       -0.17958984e-01, -0.04688417e-01,  1.76739121e-01, -1.46468375e-01,
       2.26562380e-01,  9.7652980e-01, -2.67578125e-01, -1.29832150e-01,
       1.24511739e-01, -2.23632812e-01, -2.1397188e-01,  3.19855354e-02,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       3.22265625e-01,  3.14453125e-01, -1.18116408e-01,  8.0071250e-02,
       -0.6679875e-02, -6.91339908e-01, -1.76719336e-01, -1.46468375e-01,
       9.19151625e-02,  1.07914516e-01, -1.10515162e-01, -0.34669389e-02,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       2.499675e-01, -7.1299625e-02,  2.83293125e-01, -2.75598055e-01,
       -0.466075e-02,  3.00087712e-01, -2.00087712e-01, -5.05717004e-02,
       2.02836712e-02, -1.00087712e-01,  2.77734770e-01,
       -0.55899375e-01, -7.1299625e-02,  2.83293125e-01, -2.75598055e-01,
       3.47978125e-01, -7.1299625e-02,  5.73779469e-02, -2.74658203e-02,
       -0.57479783e-01, -1.37995312e-01, -3.88133954e-02,  1.57225522e-01,
       -0.92382312e-01, -1.32012508e-01, -5.05888231e-02, -1.66015505e-01,
       1.5527538e-01,  1.6098962e-01, -1.6098962e-01, -1.49414862e-02,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       3.39160556e-02,  5.12995312e-02,  9.7652980e-02,  3.14941486e-02,
       2.51464644e-02, -2.05156250e-01, -1.24623438e-01, -6.05476552e-02,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
       -0.61230469e-02, -3.01511672e-02,  8.66099219e-03, -1.30859375e-01,
       3.0859375e-02,  3.33779862e-02, -0.09971938e-02, -0.04669389e-02,
       -1.1474694e-01, -1.47795976e-02,  9.71679688e-02, -1.66692193e-01,
       3.17734770e-01, -1.70884388e-01,  7.61718758e-02, -1.54682308e-01,
       0.12238662e-01,  0.12238662e-01,  0.12238662e-01,  0.12238662e-01,
```

Gambar 5.14 Data clear

```
In [9]: model_dirgza['shine']

Out[9]:
array([-0.13482344,  0.25976562, -0.15017969, -0.27734375,  0.30273435,
       0.09969838,  0.35925712, -0.22549219, -0.18539575,  0.3671875,
      -0.18582734,  0.13671875,  0.2598625,  0.87128986,  0.28539862,
      -0.18582734,  0.13671875,  0.2598625,  0.87128986,  0.28539862,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.20631531,  0.38805938,  0.08162586, -0.056922597,  0.14648458,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.24121894, -0.18945112, -0.15234375, -0.05493164,  0.81434326,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.24023486,  0.36132062, -0.12549219, -0.13183954,  0.24513119,
      -0.0246556,  0.32226562,  0.11579655,  0.18164602,  0.04931644,
      -0.14257812,  0.14941486,  0.08513215,  0.85498904,  0.05120539,
      -0.14257812,  0.14941486,  0.08513215,  0.85498904,  0.05120539,
      -0.01567188,  0.26989458,  0.07801328, -0.0859375,  0.05955988,
      -0.14257812,  0.14941486,  0.08513215,  0.85498904,  0.05120539,
      -0.04768742,  0.08537391, -0.04722461,  0.05908283,  0.7218986,
      0.0153775, -0.11621954,  0.07128986,  0.01468899, -0.18644531,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.3359975, -0.187,  0.14598781,  0.08843867,  0.87617188,
      -0.09521484,  0.08843867,  0.20117188,  0.11238469, -0.33984775,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.22949219,  0.14941486,  0.10531215,  0.85498904,  0.050753784,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.08740234,  0.1895809,  0.1147468,  0.06516485,  0.12618888,
      -0.11238469,  0.08808109,  0.08591406,  0.83988594,  0.0502935,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.08823975, -0.15234375,  0.19042369,  0.04988469,  0.25,
      -0.00671661,  0.04589844,  0.05182359,  0.04952754,  0.05491111,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.0291416,  0.26367188, -0.28718938,  0.09863281, -0.99862831,
      -0.03125, -0.13769531, -0.05737305,  0.38867188, -0.421875,
```

Gambar 5.15 Data shine

```
In [11]: model_dirgza['car']

Out[11]:
array([-0.13482343,  0.00942358,  0.03344272, -0.95631769,  0.04839396,
       0.1253775, -0.04951641, -0.08945351, -0.20894885,  0.11962051,
      -0.1866446, -0.25,  0.10480591, -0.18742188, -0.01879883,
      -0.05208193,  0.08821075,  0.06445312,  0.14453125, -0.04518016,
      -0.05208193,  0.08821075,  0.06445312,  0.14453125, -0.04518016,
      -0.04467773, -0.15527344,  0.2598625,  0.3398475,  0.00756836,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.09912189,  0.15698398,  0.08847861,  0.19045112,  0.02832031,
      -0.0534646,  0.03863965,  0.11639664,  0.24121894, -0.234375,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.26010531,  0.37695312, -0.12558599,  0.11429781,  0.17675781,
      -0.18007761,  0.0039304,  0.26757912,  0.20117188,  0.07119038,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.26171875, -0.08542578,  0.02583801, -0.05834961, -0.08737354,
      -0.11767578, -0.04429675,  0.17285156,  0.04394531, -0.23867575,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.32631188, -0.04492188, -0.11429781,  0.22851562, -0.01647949,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.1815625,  0.08831771,  0.18793815, -0.24689375, -0.189375,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.09375, -0.0162335,  0.20214544,  0.23144531, -0.05444336,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.17578125, -0.02779998,  0.2010156,  0.02392978,  0.03125,
      -0.25398625, -0.125,  0.05093164, -0.17582812,  0.28515625,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.00769843,  0.20597912, -0.01789894, -0.12988281,  0.04711914,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.04722461,  0.15682912,  0.15859896, -0.06654688,  0.19435594,
      -0.05883789,  0.0612739, -0.01549818,  0.87617188,  0.05120539,
      -0.07324219,  0.16895156,  0.04585988, -0.17675781, -0.37798662,
      -0.22558594,  0.16380594,  0.05102539, -0.08251953,  0.07958964,
```

Gambar 5.16 Data bag

```
In [12]: model_dirgza['wash']

Out[12]:
array([-0.46044222e-03, -0.41081562e-01, -0.56687989e-02,  1.34765625e-01,
       -0.3828159e-01,  3.24218759e-01, -0.44728562e-02, -1.29832612e-01,
       -0.1253775, -0.04951641, -0.08945351, -0.20894885,  0.11962051,
      -0.77954181e-02,  2.80897812e-01, -0.27246894e-02, -0.95668758e-01,
      -0.24187508e-02,  3.04867598e-01,  2.11014682e-01, -0.88671758e-02,
      -0.29612112e-02,  0.62189755e-01,  1.9359975e-01,  0.21728515e-02,
      -0.24187508e-02, -0.8087155e-02,  1.0909755e-01, -0.17644835e-01,
      -0.8078155e-02, -3.80897575e-02, -1.0909755e-01, -0.17644835e-01,
      1.74804688e-01,  0.78747935e-02,  1.11238125e-01,  1.0599896e-01,
      -0.17644835e-01, -0.8078155e-02,  1.0909755e-01, -0.17644835e-01,
      -0.43750898e-01, -0.12089781e-01,  3.80897575e-02, -0.50517904e-02,
      -0.24187508e-02, -2.29492188e-01, -0.21586488e-02,  4.51601556e-02,
      -0.56445312e-02,  1.77734755e-01,  0.12287012e-01, -0.71939758e-02,
      -0.24187508e-02,  0.32226562e-02, -0.92578155e-01,  1.74804688e-01,
      -0.17578008e-01,  0.32226562e-02, -0.92578155e-01,  1.74804688e-01,
      -0.17578008e-01, -0.29612112e-02,  2.05978125e-02,  2.12197988e-01,
      -0.18867175e-01,  0.32226562e-02, -0.92578155e-01,  1.74804688e-01,
      -0.58867139e-02,  0.39078125e-02,  1.66843122e-01,  2.79541816e-02,
      -0.18867175e-01,  0.32226562e-02, -0.92578155e-01,  1.74804688e-01,
      -0.69677734e-02,  1.69921797e-01, -1.46484756e-01,  2.65625088e-01,
      -0.17205356e-02,  0.12394605e-02, -1.14257812e-01,  7.22656259e-02,
      -0.58144531e-02,  0.54443394e-02,  3.79468755e-01,  5.62508808e-01,
      -0.53520312e-01, -0.37580898e-01,  2.59755256e-01, -1.49414628e-01,
      -0.66468258e-02,  2.13867188e-02, -2.86865324e-02, -1.70898438e-01,
      -0.53520312e-01, -0.37580898e-01,  2.59755256e-01, -1.49414628e-01,
      -0.17665759e-02, -2.41699218e-02, -1.77734755e-01,  2.71484375e-01,
```

Gambar 5.17 Data car

```
In [13]: model_dirga['motor']
Out[13]:
array([-2.73738669e-02, 1.5099862e-02, -4.6342578e-02, -1.3281259e-01,
       -2.5976525e-01, -1.77734575e-01, 3.68652344e-02, -4.3798000e-01,
       2.3475980e-02, 2.5781259e-01, 1.74894689e-01, 2.44348625e-02,
       -2.37714775e-01, 2.10363455e-01, 3.58142675e-01, 1.25557765e-02,
       -3.83980181e-02, 1.5628315e-01, 5.8957358e-02, 1.1284668e-01,
       1.2984715e-01, 1.6992175e-01, -1.5527435e-01, 4.58954375e-01,
       2.0773475e-01, 1.53228312e-01, 1.69921875e-01, -1.03874219e-01,
       -1.00073195e-01, 3.6447255e-01, 5.0000000e-01, 1.28233075e-01,
       1.5426875e-01, 8.88671875e-02, 2.3632815e-01, 5.6153438e-03,
       -1.0000000e-01, 3.6447255e-01, 5.0000000e-01, 1.28233075e-01,
       -3.0859577e-01, -6.50624414e-01, 6.3476525e-02, 4.02820201e-02,
       3.1080594e-02, -6.50624414e-01, 6.3476525e-02, 4.02820201e-02,
       1.5098025e-01, -1.3769512e-01, 9.96803750e-01, 1.2896250e-01,
       -3.4472556e-01, -1.8088027e-02, 2.14847350e-01, -9.3454358e-02,
       -9.0000000e-01, 2.33598438e-01, 5.05371604e-02, -3.3798625e-01,
       -2.21679888e-01, 2.33598438e-01, 5.05371604e-02, -3.3798625e-01,
       1.12813215e-01, 1.14257812e-01, -9.71679658e-02, -2.61373875e-01,
       -8.00781250e-02, -1.14257812e-01, -9.71679658e-02, -2.61373875e-01,
       3.8475525e-01, -1.8798000e-01, 1.18515152e-01, 1.08655959e-01,
       1.10865959e-01, 1.8798000e-01, 1.18515152e-01, 1.08655959e-01,
       -2.4023475e-01, 8.3466938e-02, 4.19921875e-02, -1.9185891e-02,
       9.0000000e-01, 3.27128125e-02, 5.71239902e-02, -1.9185891e-02,
       2.03125288e-02, -3.2812159e-01, 3.27148415e-02, 5.71239902e-02,
       1.7773475e-01, -9.57831259e-02, 2.45171289e-01, 6.68476552e-02,
       3.1080594e-02, -9.57831259e-02, 2.45171289e-01, 6.68476552e-02,
       4.00598025e-01, 6.69148025e-02, 3.08751250e-01, -1.99218750e-01,
       -1.23454358e-01, -1.8088027e-02, 2.14847350e-01, -9.3454358e-02,
       7.70513221e-02, 7.8089712e-02, -2.27398022e-01, -3.02754375e-01,
```

Gambar 5.18 Data wash

```
In [14]: model_dirga['cycle']
Out[14]:
array([ 0.04541016,  0.21679688, -0.02789961,  0.12353516, -0.20780315,
       -0.12813215,  0.26367188, -0.12989625, -0.125
       , 0.15328211, 0.12989625, -0.125, -0.12813215,
       -0.02563477, -0.07585395, -0.0625
       , 0.84614258, -0.10854688,
       0.8475525e-01, -0.11669922, -0.3357595
       , 0.078125
       , 0.08847266, 0.8475525e-01, -0.11669922, -0.3357595
       , 0.078125
       , 0.18902734, 0.02172852, -0.10893359, 0.82698234, -0.18644531,
       -0.18644531, 0.02172852, -0.10893359, 0.82698234, -0.18644531,
       -0.12787399, -0.81657711, -0.01967777, 0.13671875, 0.13646825,
       0.11425781, 0.26989781, -0.06979102, -0.07275391, 0.1589862,
       -0.06872383, 0.03564453, -0.29828212, 0.89969938, -0.14484375,
       -0.06767199, 0.05957931, -0.05175776, 0.19620894, 0.2265426,
       0.09963281, 0.02038374, -0.08789062, -0.07226562, -0.09423828,
       -0.17809344, 0.1484375
       , 0.18468765, 0.86445312, 0.12989625,
       -0.17809344, 0.1484375
       , 0.18468765, 0.86445312, 0.12989625,
       0.37109357, -0.15722656, 0.89126172, 0.34969038, -0.00081553,
       -0.04335489, 0.1484375
       , 0.18468765, 0.86445312, 0.12989625,
       -0.14355489, 0.83173828, -0.07421875, 0.34979588, 0.148625
       , 0.08433548, -0.12896252, 0.34969038, -0.82929698, -0.19628986,
       -0.08433548, -0.12896252, 0.34969038, -0.82929698, -0.19628986,
       -0.18931562, -0.13058468, 0.02844238, 0.18408031, 0.17773438,
       0.06894553, -0.1796875
       , 0.02781205, 0.15625
       , 0.02820567,
       0.09963281, 0.02038374, -0.08789062, -0.07226562, -0.09423828,
       0.07959864, 0.01989746, 0.25589538, 0.13778986, 0.8259862,
       -0.28808781, 0.18408031, 0.18408031, 0.18408031, 0.18408031,
       -0.28808781, 0.18408031, 0.18408031, 0.18408031, 0.18408031,
       -0.08979659, -0.22265625, -0.10498847, 0.24121894, 0.8638275,
       -0.2345791, -0.12031159, -0.12989625, 0.85444356, -0.14744864,
       -0.09960938, -0.17809344, -0.09779889, 0.35323255, 0.68843485,
```

Gambar 5.19 Data motor

```
In [15]: model_dirga.similarity('wash', 'clear')
Out[15]: 0.09019176
```

Gambar 5.20 Data cycle

```
1 model_dirga.similarity('wash', 'clear')
2 #%%
3 model_dirga.similarity('bag', 'love')
4 #%%
5 model_dirga.similarity('motor', 'car')
6 #%%
7 model_dirga.similarity('sick', 'faith')
8 #%%
9 model_dirga.similarity('cycle', 'shine')
```

Merupakan persentase dari perbandingan kata:

```
In [16]: model_dirga.similarity('bag', 'love')
Out[16]: 0.07536096
```

Gambar 5.21 Data wash dan clear

```
In [17]: model_dirga.similarity('motor', 'car')
Out[17]: 0.4810173
```

Gambar 5.22 Data bag dan love

```
In [18]: model_dirga.similarity('sick', 'faith')
Out[18]: 0.123073205
```

Gambar 5.23 Data sick dan faith

```
In [16]: model_dirga.similarity('bag', 'love')
Out[16]: 0.07536096
```

Gambar 5.24 Data cycle dan shine

5.2.2.2 Nomor 2

```
1 import re
2
3 test_string = "Dirga Brajamusti, adalah nama aku"
4 print ("Faktanya: " + test_string)
5 res = re.findall(r'\w+', test_string)
6 print ("The list of words is : " + str(res))
7 #%%
```

Kode ini berguna untuk membuat string memakai import re, dengan memakai test_string sebagai datanya

```
In [23]: import re
...
...: test_string = "Dirga Brajamusti, adalah nama aku"
...: print ("Faktanya: " + test_string)
...: res = re.findall(r'\w+', test_string)
...: print ("The list of words is : " + str(res))
Faktanya: Dirga Brajamusti, adalah nama aku
The list of words is : ['Dirga', 'Brajamusti', 'adalah', 'nama', 'aku']
```

Gambar 5.25 Nomor 2

```
1 import random
2 sent_matrix = [ [ 'Ini' , 'Data' ] , [ 'Untuk' , 'Merandom' ] , [ 'Isi' , '
  Yang' ] , [ 'Ada' , 'Disini' ] ]
3 result = ""
```

```

4 for elem in sent_matrix:
5     result += random.choice(elem) + " "
6 print (result)

```

Kode ini berguna untuk membuat string dengan import random, memakai variable sent_matrix untuk membuat string, dan result sebagai print dari random data yang diacak, hasilnya akan berubah-ubah:

```

In [23]: import re
...
... text_string = "Dirga Brajamusti, adalah nama aku"
... print ("Faktanya: " + text_string)
...
... res = findall(r'\w+', text_string)
... print ("The list of words is : " + str(res))
Faktanya: Dirga Brajamusti, adalah nama aku
The list of words is : ['Dirga', 'Brajamusti', 'adalah', 'nama', 'aku']

```

Gambar 5.26 Nomor 2-1

5.2.2.3 Nomor 3

```

1 from gensim.models.doc2vec import Doc2Vec, TaggedDocument
2 ## Exapmple document (list of sentences)
3 doc = ["I love machine learning",
4         "I love coding in python",
5         "This is a good pc",
6         "This is a good mac",
7         "This is a good phone"]
8 tokenized_doc = ['love']
9 tokenized_doc
10
11 print(doc)

```

Fungsi dari library gensim untuk pemodelan topik unsupervised learning. Fungsi dari doc2vec itu sendiri adalah untuk membandingkan bobot data yang terdapat pada dokumen lainnya, apakah kata-katanya ada yang sama atau tidak. Lalu untuk tagged document itu memasukan kata-kata pada setiap dokumennya untuk di vektorisasi, dan model untuk membuat model dan save file model.

```

In [24]: from gensim.models.doc2vec import Document, TaggedDocument
...
... doc = ["I love machine learning",
...         "I love coding in python",
...         "This is a good pc",
...         "This is a good mac",
...         "This is a good phone"]
...
... tokenized_doc = ['love']
...
... print(doc)
...
... print(tokenized_doc)
['I love machine learning', 'I love coding in python', 'This is a good pc', 'This is a good mac', 'This is a good phone']
['love']

```

Gambar 5.27 Nomor 3

```

1 tagged_data = [TaggedDocument(d, [i]) for i, d in enumerate(
    tokenized_doc)]
2 tagged_data
3 ## Train doc2vec model
4 model = Doc2Vec(tagged_data, vector_size=20, window=2, min_count
    =1, workers=4, epochs = 100)
5 # Save trained doc2vec model
6 model.save("test_doc2vec.model")
7 ## Load saved doc2vec model
8 model=Doc2Vec.load("test_doc2vec.model")
9 ## Print model vocabulary
10 model.wv.vocab

```

```
In [8]: from gensim.models.doc2vec import Doc2Vec, TaggedDocument

# Create Doc2Vec documents
docs = ["I love machine learning",
        "I love this machine learning",
        "This is a good pc",
        "This is a good mac",
        "I don't like windows"]
tagged_docs = []
for i, doc in enumerate(docs):
    tagged_docs.append(TaggedDocument(doc, [i]))
print(tagged_docs)

# Train the Doc2Vec model
model = Doc2Vec(tagged_docs, vector_size=10, window=2, min_count=1, workers=4)
model.train(tagged_docs, total_examples=len(tagged_docs), epochs=100)

# Test the Doc2Vec model
print(model.infer_vector(["I", "love", "this", "mac"]))

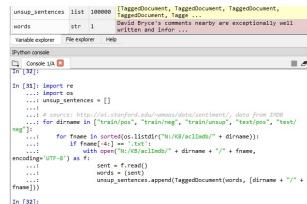
# Print the first 5 words
print(model.wv.most_similar("love"))
```

Gambar 5.28 Nomor 3-1

5.2.2.4 Nomor 4

```
1 import re
2 import os
3 unsup_sentences = []
4
5 # source: http://ai.stanford.edu/~amaas/data/sentiment/, data
6 # from IMDB
7 for dirname in ["train/pos", "train/neg", "train/unsup", "test/
8 pos", "test/neg"]:
9     for fname in sorted(os.listdir("N:/KB/aclImdb/" + dirname)):
10         if fname[-4:] == '.txt':
11             with open("N:/KB/aclImdb/" + dirname + "/" + fname,
12 encoding='UTF-8') as f:
13                 sent = f.read()
14                 words = (sent)
15                 unsup_sentences.append(TaggedDocument(words, [
16                     dirname + "/" + fname]))
```

Disini memakai dataset dari aclImdb. Untuk menambahkan data training kita melakukan import library os, library os fungsinya untuk melakukan interaksi antara python dengan windows atau os kita, setelah itu buat variable unsup sentences. Lalu pilih direktori tempat data akan disimpan. Lalu untuk mensortir data yang terdapat pada folder aclImdb dan membaca file tersebut dengan ekstensi .txt. Hasil dari code pertama tersebut adalah adanya data hasil running dari folder aclImdb.



Gambar 5.29 Nomor 4

5.2.2.5 Nomor 5

```
1 #Pengacakan data
2 mute = (unsup_sentences)
3
4 #Pembersihan data
5 model.delete_temporary_training_data(keep_inference=True)
```

Pada bagian pengacakan data berguna untuk mengacak data agar saat data di running bisa berjalan lebih baik dan hasil presentase akan lebih baik. Sedangkan untuk pembersihan data untuk memberikan ruang bagi ram laptop kita setelah melakukan running data lebih dari 3 juta , agar lebih ringan saat proses selanjutnya.

```
In [32]: mute = (unsup_sentences)
...+
...+ @membership_data
...+ model.delete_temporary_training_data(keep_inference=True)
```

Gambar 5.30 Nomor 5

5.2.2.6 Nomor 6

```
1 #Save data
2 model.save('dirga.d2v')
3
4 #Delete temporary data
5 model.delete_temporary_training_data(keep_inference=True)
```

Save berfungsi untuk menyimpan hasil dari proses pelatihan data sebelumnya ke dalam sebuah file, model tersebut dilakukan penyimpanan untuk memberikan keringanan pada ram agar saat akan melakukan pelatihan lagi, model tersebut bisa load tanpa harus melakukan pelatihan dari awal dan akan menghemat waktu. Sedangkan untuk delete temporary training data ini berfungsi untuk menghapus data latihan yang sebelumnya sudah dilakukan dan disimpan, bertujuan untuk memberikan keringanan pada ram. Karena setelah melakukan proses pelatihan ram biasanya jadi penuh sampai komputer menjadi lag. Itulah fungsi dari delete temporary training data

```
In [33]: model.save('dirga.d2v')
...+
...+ @memberships
...+ model.delete_temporary_training_data(keep_inference=True)
2020-04-05 21:40:22,528 : INFO : Saving DocVec object under dirga.d2v, separately None
2020-04-05 21:40:22,528 : INFO : saved dirga.d2v
```

Gambar 5.31 Nomor 6

5.2.2.7 Nomor 7

```
1 model.infer_vector(res)
```

Infer_vector berfungsi untuk membandingkan kata yang tercantum dengan vektor yang mana pada dokumen yang sudah diload pada step sebelumnya. Selain itu infer_vector juga untuk menghitung atau mengkalkulasikan vektor dari kata yang dicantumkan dari model yang telah dibuat. Alangkah baiknya kata yang dicantumkan itu lebih panjang lagi agar hasilnya bisa lebih baik lagi.

```
In [79]: model.infer_vector(res)
Out[79]:
array([-0.82322483, -0.0122747, -0.00295202, -0.00356266, -0.02363259,
       -0.00529805,  0.00301624, -0.00417492, -0.00891156,  0.00759805,
       0.01598096, -0.00897159, -0.01558069,  0.01741262,  0.01701254,
       0.0122115,  0.01433158,  0.00034627,  0.01111711,  0.0143115],  
dtype='float32')
```

Gambar 5.32 Nomor 7

5.2.2.8 Nomor 8

```
1 from sklearn.metrics.pairwise import cosine_similarity
2 cosine_similarity(
3     [model.infer_vector(["datanya", "banyak"]),
4      [model.infer_vector(["pusing", "data"])]])
```

Cosine_similarity berfungsi untuk membandingkan vektorisasi data diantara kedua kata yang di inputkan, jika hasil presentase dari kedua kata tersebut lebih dari 50% itu memiliki kemungkinan kata tersebut terdapat dalam 1 file. Namun jika kurang dari 50% itu kemungkinan kata tersebut tidak terdapat dalam 1 file. Hasil yang didapatkan pada code tersebut hanya 0.49% itu dikarenakan kata pertama dan kedua tidak memiliki kesamaan vektorisasi dan tidak terdapat pada salah satu dokumen.

```
In [83]: from sklearn.metrics.pairwise import cosine_similarity
... cosine_similarity(
...     [model.infer_vector(["datanya", "banyak"]),
...      [model.infer_vector(["pusing", "data"])]])
Out[83]: array([-0.490003132]), dtype=float32)
```

Gambar 5.33 Nomor 8

5.2.2.9 Nomor 9

```
1 from sklearn import datasets, linear_model
2 from sklearn.model_selection import cross_val_score
3 diabetes = datasets.load_diabetes()
4 X = diabetes.data[:150]
5 y = diabetes.target[:150]
6 lasso = linear_model.Lasso()
7 print(cross_val_score(lasso, X, y, cv=3))
```

Melakukan perhitungan presentase dengan menggunakan cross_validation dengan metode kneighborsClassifier. Menggunakan dataset iris

```
In [78]: from sklearn import datasets, linear_model
... from sklearn.model_selection import cross_val_score
... diabetes = datasets.load_diabetes()
... X = diabetes.data[:150]
... y = diabetes.target[:150]
... lasso = linear_model.Lasso()
... print(cross_val_score(lasso, X, y, cv=3))
[0.33150734 0.08022311 0.03531764]
```

Gambar 5.34 Nomor 9

5.2.3 Penanganan Error

- Error

1. FileNotFoundException

```
FileNotFoundException: [Errno 2] No such file or directory: 'N:/E8a/GoogleNews-vectors-negative3B8.bin'
```

Gambar 5.35 Error 1

- Solusi

1. FileNotFoundException

Perhatikan letak file ada dimana, pastikan path telah benar

5.2.4 Bukti Tidak Plagiat



Gambar 5.36 Bukti Tidak Plagiat

5.2.5 Link Youtube

<https://youtu.be/AtLNMwxCUSk>

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.