

**CERDAS MENGUASAI GIT**



---

# CERDAS MENGUASAI GIT

## Dalam 24 Jam

---

**Rolly M. Awangga**  
Informatics Research Center



**Kreatif Industri Nusantara**

***Penulis:***

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

***Editor:***

M. Yusril Helmi Setyawan

***Penyunting:***

Syafrial Fachrie Pane

Khaera Tunnisia

Diana Asri Wijayanti

***Desain sampul dan Tata letak:***

Deza Martha Akbar

***Penerbit:***

Kreatif Industri Nusantara

***Redaksi:***

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

***Distributor:***

Informatics Research Center

Jl. Sariasisih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara  
apapun tanpa ijin tertulis dari penerbit

*'Jika Kamu tidak dapat  
menahan lelahnya  
belajar, Maka kamu harus  
sanggup menahan  
perihnya Kebodohan.'*

*Imam Syafi'i*

## CONTRIBUTORS

---

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia



# CONTENTS IN BRIEF

---

<b>1 Chapter 1</b>	<b>1</b>
<b>2 Chapter 2</b>	<b>91</b>
<b>3 Chapter 3</b>	<b>159</b>
<b>4 Chapter 4</b>	<b>161</b>
<b>5 Chapter 5</b>	<b>163</b>
<b>6 Chapter 6</b>	<b>165</b>
<b>7 Chapter 7</b>	<b>167</b>



# DAFTAR ISI

---

Daftar Gambar	xi
Daftar Tabel	xiii
Foreword	xvii
Kata Pengantar	xix
Acknowledgments	xxi
Acronyms	xxiii
Glossary	xxv
List of Symbols	xxvii
Introduction	xxix
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
<b>1 Chapter 1</b>	<b>1</b>
1.1 1174006 - Kadek Diva Krishna Murti	1
1.1.1 Teori	1
1.1.2 Praktek	9
1.1.3 Penanganan Error	13

1.1.4	Bukti Tidak Plagiat	15
1.2	1174027 - Harun Ar - Rasyid	16
1.2.1	Teori	16
1.2.2	Praktek	17
1.2.3	Penanganan Error	19
1.2.4	Bukti Tidak Plagiat	20
1.3	1174031 - Muhammad Tomy Nur Maulidy	20
1.3.1	Teori	20
1.3.2	Praktek	21
1.3.3	Penanganan Error	23
1.3.4	Bukti Tidak Plagiat	24
1.4	1174003 - Dwi Septiani Tsaniyah	25
1.4.1	Teori	25
1.4.2	Praktek	26
1.4.3	Penanganan Error	28
1.4.4	Bukti Tidak Plagiat	29
1.5	1174051 - Evietania Charis Sujadi	29
1.5.1	Teori	29
1.5.2	Instalasi	30
1.5.3	Penanganan Error	32
1.5.4	Bukti Tidak Plagiat	33
1.6	1174021 - Muhammad Fahmi	33
1.6.1	Teori	33
1.6.2	Praktek	35
1.6.3	Penanganan Error	37
1.6.4	Bukti Tidak Plagiat	38
1.7	1174096 - Nico Ekklesia Sembiring	38
1.7.1	Teori	38
1.7.2	Praktek	40
1.7.3	Penanganan Error	42
1.7.4	Bukti Tidak Plagiat	43
1.8	1174012 - Damara Benedikta	43
1.8.1	Teori	43
1.8.2	Praktek	44
1.8.3	Penanganan Error	46
1.8.4	Bukti Tidak Plagiat	47
1.9	1174095 - Muhammad Dzihan Al-Bannai	47
1.9.1	Teori	47

1.9.2	Praktek	49
1.9.3	Penanganan Error	51
1.9.4	Bukti Tidak Plagiat	51
1.10	1174009 - Dwi Yulianingsih	52
1.10.1	Teori	52
1.10.2	Praktek	53
1.10.3	Penanganan Error	55
1.10.4	Bukti Tidak Plagiat	56
1.11	1174008 - Arjun Yuda Firwanda	56
1.11.1	Teori	56
1.11.2	Praktek	57
1.11.3	Penanganan Error	59
1.11.4	Bukti Tidak Plagiat	60
1.12	1174005 - Oniwaldus Bere Mali	60
1.12.1	Teori	60
1.12.2	Praktek	62
1.12.3	Penanganan Error	63
1.12.4	Bukti Tidak Plagiat	64
1.13	1174026- Felix Setiawan Lase	64
1.13.1	Teori	64
1.13.2	Praktek	66
1.13.3	Penanganan Error	68
1.13.4	Bukti Tidak Plagiat	68
1.14	1174004 - Choirulanam	69
1.14.1	Teori	69
1.14.2	Praktek	70
1.14.3	Penanganan Error	72
1.14.4	Bukti Tidak Plagiat	73
1.15	1164013 - Ikrima Ningrum Sari Mulyana	73
1.15.1	Teori	73
1.15.2	Praktek	75
1.15.3	Penanganan Error	77
1.15.4	Bukti Tidak Plagiat	77
1.16	Sri Rahayu - 1174015	78
1.16.1	Pemahaman Teori	78
1.16.2	Intalasi	79
1.16.3	Penanganan Error	81
1.16.4	Bukti Tidak Melakukan Plagiat	82

1.16.5	Link Youtube	82
1.17	1174002 - Habib Abdul Rasyid	82
1.17.1	Teori	82
1.17.2	Praktek	84
1.17.3	Penanganan Error	89
1.17.4	Bukti Tidak Plagiat	89

<b>2</b>	<b>Chapter 2</b>	<b>91</b>
2.1	1174006 - Kadek Diva Krishna Murti	91
2.1.1	Teori	92
2.1.2	Praktek	92
2.1.3	Penanganan Error	92
2.1.4	Bukti Tidak Plagiat	92
2.2	1174027 - Harun Ar - Rasyid	92
2.2.1	Teori	92
2.2.2	Praktek	97
2.2.3	Penanganan Error	99
2.2.4	Bukti Tidak Plagiat	100
2.3	1174012 - Damara Benedikta	100
2.3.1	Teori	100
2.3.2	Praktek	105
2.3.3	Penanganan Error	107
2.3.4	Bukti Tidak Plagiat	108
2.4	1174021 - Muhammad Fahmi	108
2.4.1	Teori	108
2.4.2	Praktek	113
2.4.3	Penanganan Error	119
2.4.4	Bukti Tidak Plagiat	121
2.5	1174031 - Muhammad Tomy Nur Maulidy	121
2.5.1	Teori	121
2.5.2	Praktek	126
2.5.3	Penanganan Error	127
2.5.4	Bukti Tidak Plagiat	129
2.6	1174051 - Evietania Charis Sujadi	129
2.6.1	Teori	129
2.6.2	Praktek	134
2.6.3	Penanganan Error	135
2.6.4	Bukti Tidak Plagiat	136

2.7	1174095 - Muhammad Dzihan Al-Banna	137
2.7.1	Teori	137
2.7.2	Praktek	141
2.7.3	Penanganan Error	148
2.7.4	Bukti Tidak Plagiat	149
2.8	1174026 - Felix Setiawan Lase	149
2.8.1	Teori	149
2.8.2	Praktek	154
2.8.3	Penanganan Error	155
2.8.4	Bukti Tidak Plagiat	157
<b>3</b>	<b>Chapter 3</b>	<b>159</b>
3.1	1174006 - Kadek Diva Krishna Murti	159
3.1.1	Teori	160
3.1.2	Praktek	160
3.1.3	Penanganan Error	160
3.1.4	Bukti Tidak Plagiat	160
<b>4</b>	<b>Chapter 4</b>	<b>161</b>
4.1	1174006 - Kadek Diva Krishna Murti	161
4.1.1	Teori	162
4.1.2	Praktek	162
4.1.3	Penanganan Error	162
4.1.4	Bukti Tidak Plagiat	162
<b>5</b>	<b>Chapter 5</b>	<b>163</b>
5.1	1174006 - Kadek Diva Krishna Murti	163
5.1.1	Teori	164
5.1.2	Praktek	164
5.1.3	Penanganan Error	164
5.1.4	Bukti Tidak Plagiat	164
<b>6</b>	<b>Chapter 6</b>	<b>165</b>
6.1	1174006 - Kadek Diva Krishna Murti	165
6.1.1	Teori	166
6.1.2	Praktek	166
6.1.3	Penanganan Error	166
6.1.4	Bukti Tidak Plagiat	166

<b>7 Chapter 7</b>	<b>167</b>
7.1 1174006 - Kadek Diva Krishna Murti	167
7.1.1 Teori	168
7.1.2 Praktek	168
7.1.3 Penanganan Error	168
7.1.4 Bukti Tidak Plagiat	168

# **FOREWORD**

---

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa



# KATA PENGANTAR

---

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

*Bandung, Jawa Barat*

*Februari, 2019*



## ACKNOWLEDGMENTS

---

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Intership.

R. M. A.



## ACRONYMS

---

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association



## GLOSSARY

---

git	Merupakan manajemen sumber kode yang dibuat oleh linus torvald.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald



# SYMBOLS

---

$A$  Amplitude

$\&$  Propositional logic symbol

$a$  Filter Coefficient

$B$  Number of Beats



# INTRODUCTION

---

ROLLY MAULANA AWANGGA, S.T., M.T.

Informatics Research Center  
Bandung, Jawa Barat, Indonesia

Pada era disruptif saat ini. git merupakan sebuah kebutuhan dalam sebuah organisasi pengembangan perangkat lunak. Buku ini diharapkan bisa menjadi penghantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[?].

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta \sum_{def}^{abc} \quad (I.1)$$



# **BAB 1**

---

# **CHAPTER 1**

---



## **BAB 2**

---

## **CHAPTER 2**

---



## BAB 3

---

# CHAPTER 3

---

### 3.1 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 rehabilitation  
3   program using AES-CBC URL encryption and QR-Code},  
4   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and  
5   Hasanudin, Trisna Irmayadi},  
6   booktitle={Information Technology, Information Systems and  
7   Electrical Engineering (ICITISEE), 2017 2nd International  
8   conferences on},  
9   pages={255--260},  
10  year={2017},  
11  organization={IEEE}  
12 }
```



**Gambar 3.1** 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.

### **3.1.1 Teori**

### **3.1.2 Praktek**

### **3.1.3 Penanganan Error**

### **3.1.4 Bukti Tidak Plagiat**



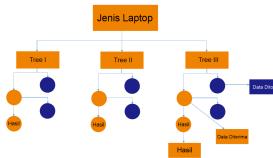
**Gambar 3.2** Kecerdasan Buatan.

## **3.2 1174021 - Muhammad Fahmi**

### **3.2.1 Soal Teori**

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest merupakan 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. Untuk gambar Random Forest dapat dilihat dibawah ini :



**Gambar 3.3** Random Forest.

2. Jelaskan 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. Jelaskan apa itu cross validation.

Cross Validation merupakan 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 mengujinya dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh 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 decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision 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.

- Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

1 fig = plt.figure() #hasil plot sebagai figure
2 fig.clf() #figure di ambil dari clf
3 ax = fig.gca(projection='3d') #ax sebagai projection 3d
4 x = rf_params[:,0] #x sebagai index 0
5 y = rf_params[:,1] #y sebagai index 1

```

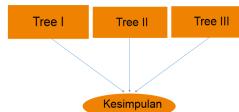
Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data posotif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data posotif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat dibawah ini :



**Gambar 3.4** Confusion Matriks.

- Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri

Voting pada random forest ialah sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk lebih jelasnya saya akan memberikan sebuah contoh bagaimana voting bekerja :



**Gambar 3.5** Decision Tree.

Dimana ditunjukkan pada gambar diatas terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan diadakan voting untuk menentukan sebuah laptop. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan laptop asus, yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah laptop asus. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menyatakan laptop asus sedangkan pada tree 3 menyatakan laptop HP, maka kesimpulan yang diambil adalah laptop asus dikarenakan hasil voting terbanyak adalah laptop asus.

### 3.2.2 Praktek Program

1. Buat aplikasi sederhana menggunakan pandas dan jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas).

```

1 # In[44]: Soal1
2
3 import pandas as fahmi #melakukan import pada library pandas
4         sebagai fahmi
5 laptop = {"Nama Laptop" : ['Asus', 'HP', 'Lenovo', 'Samsung']} # membuat varibel yang bernama laptop, dan mengisi dataframe
6         nama2 laptop
    
```

Kode di atas digunakan untuk mengimpor atau mengirim library pandas sebagai fahmi, Hasilnya adalah sebagai berikut :

	Nama Laptop
0	Fahmi Punya Laptop Asus
1	Fahmi Punya Laptop HP
2	Fahmi Punya Laptop Lenovo
3	Fahmi Punya Laptop Samsung

**Gambar 3.6** Hasil Soal 1.

2. buat aplikasi sederhana menggunakan numpy dan jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```

1 import numpy as fahmi #melakukan import numpy sebagai fahmi
2
3 matrix_x = fahmi.eye(10) #membuat matrix dengan numpy dengan
4         menggunakan fungsi eye
5 matrix_x #deklrasikan matrix_x yang telah dibuat
6
6 print (matrix_x) #print matrix_x yang telah dibuat dengan 10x10
    
```

Fungsi eye disini berguna untuk memanggil matrix identitas dengan jumlah kolom dan baris sesuai yang ditentukan. Disini saya telah menentukan 10 kolom

dan baris maka dari itu hasilnya akan memunculkan matrix identitas 10x10, Hasilnya adalah sebagai berikut :

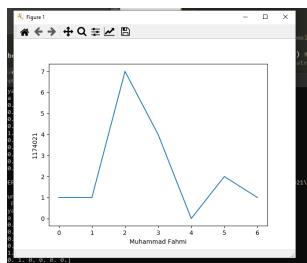
```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]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

**Gambar 3.7** Hasil Soal 2.

3. buat aplikasi sederhana menggunakan matplotlib dan jelaskan arti dari setiap baris kode(harus beda dengan teman sekelas)

```
1 import matplotlib.pyplot as fahmi #import matplotlib sebagai fahmi
2
3 fahmi.plot([1,1,7,4,0,2,1]) #memberikan nilai plot atau grafik pada fahmi
4 fahmi.xlabel('Muhammad Fahmi') #memberikan label pada x
5 fahmi.ylabel('1174021') #memberikan label pada y
6 fahmi.show() #print hasil plot berbentuk grafik
```

Jalankan source code diatas dengan spyder atau anaconda sehingga hasilnya akan seperti berikut :



**Gambar 3.8** Hasil Soal 3.

4. jalankan program klasifikasi Random Forest pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- 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.

```

1 # In [1]: RANDOM FOREST
2
3 import pandas as pd #import library pandas sebagai as
4
5 imgatt = pd.read_csv("data/CUB_200_2011/attributes/
6     image_attribute_labels.txt",
7                         sep='|', header=None, error_bad_lines=
8     False, warn_bad_lines=False,
9                         usecols=[0,1,2], names=['imgid', 'attid',
10                         'present']) #library imgatt sebagai membaca file csv dari
11 dataset, dengan ketentuan yang ada.

```

Hasilnya adalah seperti ini :

```

In [1]: Import pandas as pd
        ...
        imgatt = pd.read_csv("data/CUB_200_2011/attributes/image_attribute_labels.txt",
        ...                         sep='|', header=None, error_bad_lines=False,
        ...                         warn_bad_lines=False,
        ...                         usecols=[0,1,2], names=['imgid', 'attid',
        ...                         'present'])
        ...
        # description from dataset README
        ...
        # This file contains the attributes assigned by humans for each image
        # it is contained in the file attributes/image_attribute_labels.txt, with
        # the first column being the image id and the second column being the attribute id
        ...
        # where imgid, attid, attribute_label, and present correspond to the IDs
        # in images.txt, attributes/attribute_labels.txt, and attribute/attribute_labels.txt
        ...
        # in images.txt, attribute/attribute_labels.txt, and attribute/attribute_labels.txt
        # present is 1 if the attribute is present in the image, and 0 if it is not.
        ...
        # present). -1 means the row isn't the same as the others in the dataset.

```

**Gambar 3.9** Hasil Soal 4 - 1

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

```

1 # In [2]:
2
3 imgatt.head() #menampilkan data yang di baca tadi tetapi hanya
               data paling atas

```

Hasilnya adalah seperti ini :

	In [2]: imgatt.head()	Out[2]:
		imgid attid present
0	1	1 0
1	1	2 0
2	1	3 0
3	1	4 0
4	1	5 1

**Gambar 3.10** Hasil Soal 4 - 2

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

```

1 # In [3]:
2
3 imgatt.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

```
In [3]: imgatt.shape
Out[3]: (3677856, 3)
```

**Gambar 3.11** Hasil Soal 4 - 3

- Variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya.

```
1 # In [4]:
2
3 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values=
   'present') #membuat variabel baru dari fungsi imgatt,
   dengan mengganti index dan kolom (kebalikan)
```

Hasilnya adalah seperti ini :

```
In [4]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
```

**Gambar 3.12** Hasil Soal 4 - 4

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

```
1 # In [5]:
2
3 imgatt2.head() #menampilkan data yang di baca tadi tetapi
   hanya data paling atas
```

Hasilnya adalah seperti ini :

	(In [5]: imgatt2.head())																															
Out[5]:	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312																	
imgid	0	0	0	0	0	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	0	0	0	...	0	0	0	0	0	0	0																	
4	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0																	
5	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0																

[5 rows x 312 columns]

**Gambar 3.13** Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 # In [6]:
2
3 imgatt2.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

```
In [6]: imgatt2.shape
Out[6]: (11788, 312)
```

**Gambar 3.14** Hasil Soal 4 - 6

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

```

1 # In [7]:
2
3 imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
4                         sep=' ', header=None, names=['imgid',
5                         'label']) #baca data csv dengan ketentuan yang ada
6
7 imglabels = imglabels.set_index('imgid') #variabel imglabels
8         sebagai set index (imgid)

```

Hasilnya adalah seperti ini :

```

In [8]: imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
... sep=' ', header=None, names=['imgid', 'label'])
...
Out[8]: imglabels = imglabels.set_index('imgid')
...
In [9]: imglabels.head()
...
Out[9]: imglabels.head()
          label
imgid
1           1
2           1
3           1
4           1
5           1

```

**Gambar 3.15** Hasil Soal 4 - 7

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

```

1 # In [8]:
2
3 imglabels.head() #menampilkan data yang di baca tadi tetapi
4         hanya data paling atas

```

Hasilnya adalah seperti ini :

```

In [9]: imglabels.head()
Out[9]:
          label
imgid
1           1
2           1
3           1
4           1
5           1

```

**Gambar 3.16** Hasil Soal 4 - 8

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

```

1 # In [9]:
2
3 imglabels.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

In [10]:	imglabels.shape
Out[10]:	(11788, 1)

**Gambar 3.17** Hasil Soal 4 - 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 # In [10]:
2
3 df = imgatt2.join(imglabels) #variabel df sebagai fungsi join
    dari data imgatt2 ke variabel imglabels
4 df = df.sample(frac=1) #variabel df sebagai sample dengan
    ketentuan frac=1

```

Hasilnya adalah seperti ini :

In [11]:	df = imgatt2.join(imglabels)
...:	df = df.sample(frac=1)

**Gambar 3.18** Hasil Soal 4 - 10

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

```

1 # In [11]:
2
3 df_att = df.iloc[:, :312] #membuat kolom dengan ketentuan 312
4 df_label = df.iloc[:, 312:] #membuat kolom dengan ketentuan
    312

```

Hasilnya adalah seperti ini :

In [12]:	df_att = df.iloc[:, :312]
...:	df_label = df.iloc[:, 312:]

**Gambar 3.19** Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```

1 # In [12]:
2
3 df_att.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas

```

Hasilnya adalah seperti ini :

In [13]:	df_att.head()																																																																																																
Out[13]:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>imgId</th> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> <th>6</th> <th>7</th> <th>...</th> <th>306</th> <th>307</th> <th>308</th> <th>309</th> <th>310</th> <th>311</th> <th>312</th> </tr> </thead> <tbody> <tr><td>2449</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>2835</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>7472</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>2064</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>9834</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> </tbody> </table>	imgId	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312	2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1	2835	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	2064	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	9834	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1
imgId	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312																																																																																		
2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1																																																																																		
2835	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																		
7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1																																																																																		
2064	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																		
9834	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1																																																																																		
	[5 rows x 312 columns]																																																																																																

**Gambar 3.20** Hasil Soal 4 - 12

- Mengecek isi data teratas dari df\_label.

```
1 # In [13]:
2
3 df_label.head() #menampilkan data yang di baca tadi tetapi
      hanya data paling atas
```

Hasilnya adalah seperti ini :

	In [14]: df_label.head()
Out[14]:	label
imgid	
2449	43
988	18
7472	128
7084	121
9834	168

**Gambar 3.21** Hasil Soal 4 - 13

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

```
1 # In [14]:
2
3 df_train_att = df_att[:8000] #akan membagi 8000 row pertama
      menjadi data training dan sisanya adalah data testing
4 df_train_label = df_label[:8000] #akan membagi 8000 row
      pertama menjadi data training dan sisanya adalah data
      testing
5 df_test_att = df_att[8000:] #kebalikan dari akan membagi 8000
      row pertama menjadi data training dan sisanya adalah data
      testing
6 df_test_label = df_label[8000:] #kebalikan dari akan membagi
      8000 row pertama menjadi data training dan sisanya adalah
      data testing
7
8 df_train_label = df_train_label['label'] #menampilkan data
      training
9 df_test_label = df_test_label['label'] #menampilkan data
      testing
```

Hasilnya adalah seperti ini :

	In [15]: df_train_att = df_att[:8000] ...: df_train_label = df_label[:8000] ...: df_test_att = df_att[8000:] ...: df_test_label = df_label[8000:] ...: ...: df_train_label = df_train_label['label'] ...: df_test_label = df_test_label['label']
--	--

**Gambar 3.22** Hasil Soal 4 - 14

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

```

1 # In [15]:
2
3 from sklearn.ensemble import RandomForestClassifier #import
4         randomforestclassifier
5 clf = RandomForestClassifier(max_features=50, random_state=0,
6     n_estimators=100) #clf sebagai variabel untuk klafifikasi
7         random forest

```

Hasilnya adalah seperti ini :

**Gambar 3.23** Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 # In [16]:
2
3 clf.fit(df_train_att, df_train_label) #variabel clf untuk fit
4         yaitu training

```

Hasilnya adalah seperti ini :

**Gambar 3.24** Hasil Soal 4 - 16

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

```

1 # In [17]:
2
3 print(clf.predict(df_train_att.head())) #print clf yang di
4         prediksi dari training tetapi hanya menampilkan data
5         paling atas

```

Hasilnya adalah seperti ini :

**Gambar 3.25** Hasil Soal 4 - 17

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

```

1 # In [18]:
2
3 clf.score(df_test_att, df_test_label) #print clf sebagai
4         testing yang sudah di training tadi

```

Hasilnya adalah seperti ini :

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.4390179514255544
```

**Gambar 3.26** Hasil Soal 4 - 18

- Jalankan program confusion matrix pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Melakukan import confusion matrix pada library sklearn matriks.

```
1 # In [19]: Confusion Matrix
2
3 from sklearn.metrics import confusion_matrix #import Confusion
4 Matrix
5 pred_labels = clf.predict(df_test_att) #sebagai data testing
6 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
7 variabel data label
```

Hasilnya adalah seperti ini :

```
In [20]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

**Gambar 3.27** Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```
1 # In [20]:
2
3 cm #menampilkan data label berbentuk array
```

Hasilnya adalah seperti ini :

```
In [21]: cm
Out[21]:
array([[ 8,  1,  1, ...,  0,  1,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 0,  1,  6, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  1,  0,  0],
       [ 0,  0,  0, ...,  0,  3,  0],
       [ 0,  0,  0, ...,  0,  0, 16]], dtype=int64)
```

**Gambar 3.28** Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```
1 # In [21]:
2
3 import matplotlib.pyplot as plt #import library matplotlib
4 sebagai plt
5 import itertools #import library itertools
6 def plot_confusion_matrix(cm, classes,
```

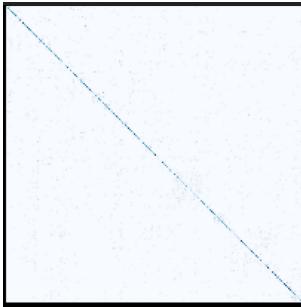
```
6                         normalize=False ,
7                         title='Confusion matrix ' ,
8                         cmap=plt.cm.Blues): #membuat fungsi
9     dengan ketentuan data yang ada pada cm
10
11     if normalize:
12         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13
14         print("Normalized confusion matrix") #jika normalisasi
15         sebagai ketentuan yang ada maka print normalized
16         confusion matrix
17     else:
18         print('Confusion matrix , without normalization') #jika
19         tidak memenuhi kondisi if maka pring else
20
21     print(cm) #print data cm
22
23
24
25     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
26     sebagai fungsi untuk membuat plot
27     plt.title(title) #membuat title pada plot
28     #plt.colorbar()
29     tick_marks = np.arange(len(classes)) #membuat marks pada
30     plot
31     plt.xticks(tick_marks, classes, rotation=90) #membuat
32     ticks pada x
33     plt.yticks(tick_marks, classes) #membuat ticks pada y
34
35     fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
36     thresh = cm.max() / 2. #variabel thresh menambil data max
37     pada cm kemudian dibagi 2
38
39
40
41     plt.tight_layout() #mengatur layout pada plot
42     plt.ylabel('True label') #memberi nama label pada sumbu y
43     plt.xlabel('Predicted label') #memberi nama label pada
44     sumbu x
45
46
47
48
49
50
51
52
53 # In [22]:
54
55
56 birds = pd.read_csv("data/CUB_200_2011/classes.txt",
57                      sep='\s+', header=None, usecols=[1], names
58                      =[ 'birdname']) #membaca csv dengan ketentuan nama birdname
59 birds = birds[ 'birdname'] #nama birds dengan ketentuan
60         birdname
61 birds #menampilkan data birds
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
```

```
47 plt.show() #menampilkan hasil plot yang berbentuk grafik
```

Hasilnya adalah seperti dibawah ini :

```
In [23]: import numpy as np
... import tensorflow as tf
... def plot_confusion_matrix(cm, classes,
...                          normalize=False,
...                          title='Confusion matrix',
...                          cmap=plt.cm.Blues):
...     """
...     This function prints and plots the confusion matrix.
...     Normalization can be applied by setting `normalize=True`.
...     """
...     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, interpolation='nearest', cmap=cmap)
...     plt.title(title)
...     plt.colorbar()
...     tick_marks = np.arange(len(classes))
...     plt.xticks(tick_marks, classes, rotation=45)
...     plt.yticks(tick_marks, classes, rotation=45)
...
...     thresh = cm.max() / 2.
...     for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
...         color = "white" if cm[i, j] > thresh else "black"
...
...         plt.text(j, i, format(cm[i, j], 'd'),
...                  horizontalalignment="center",
...                  verticalalignment="center",
...                  color=color)
...
...     plt.tight_layout()
...     plt.ylabel('True label')
...     plt.xlabel('Predicted label')
...     plt.show()

In [24]: birds = pd.read_csv("data/df_200_birds.csv")
... df_train_att = birds[0:150]
... df_train_label = birds[0:150].name
... df_test_att = birds[150:200]
... df_test_label = birds[150:200].name
... Out[24]:
0    001.Black_Footed_Albatross
1    002.Bay_Altic_Albatross
2    003.Bonaparts_Murrelet
3    004.Gannet_Billed_Auk
4    005.Gannet_Great_Auk
195   100_House_Lark
196   107_Marsh_Lark
197   108_Spoon_Billed_Auk
198   199_Winter_Lark
199   200_Cuckoo
Name: name, Length: 200, dtype: object
```



Gambar 3.29 Menampilkan hasil plot

6. Jalankan program klasifikasi SVM dan Decission Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Menunjukkan klarifikasi dengan decission tree menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```
1 # In[24]: Mencoba dengan metode Decission Tree dan SVM
2
3 from sklearn import tree #import library tree
4 clftree = tree.DecisionTreeClassifier() #clftree sebagai
5          variabel untuk decision tree
6 clftree.fit(df_train_att , df_train_label) #sebagai data
7          training
8 clftree.score(df_test_att , df_test_label) #sebagai data
9          testing
```

Hasilnya adalah seperti ini :

```
In [27]: from sklearn import tree
... clftree = tree.DecisionTreeClassifier()
... clftree.fit(df_train_att, df_train_label)
... clftree.score(df_test_att, df_test_label)
Out[27]: 0.26135163674762407
```

Gambar 3.30 Hasil Soal 6 - 1

- Menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 # In [25]:
2
3 from sklearn import svm #import library svm
4 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
5         fungsi SVC
6 clfsvm.fit(df_train_att , df_train_label) #sebagai data
7         training
8 clfsvm.score(df_test_att , df_test_label) #sebagai data testing

```

Hasilnya adalah seperti ini :

```

In [28]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
Out[28]: 0.47729672650475186

```

**Gambar 3.31** Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

```

1 # In [26]: Pengecekan Cross Validation
2
3 from sklearn.model_selection import cross_val_score #import
4         cross_val_score
5 scores = cross_val_score(clf , df_train_att , df_train_label , cv
6         =5) #variabel scores sebagai variabel prediksi dari data
7         training
8 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
9         std() * 2)) #print data scores dengan ketentuan akurasi

```

Hasilnya adalah seperti ini :

```

In [29]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: scores
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.45 (+/- 0.03)

```

**Gambar 3.32** Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

```

1 # In [27]:
2
3 scorestree = cross_val_score(clftree , df_train_att ,
4         df_train_label , cv=5) #sebagai prediksi menggunakan scores
5         dan metode tree
6 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
7         scorestree.std() * 2)) #menampilkan dengan ketentuan yang
8         ada

```

Hasilnya adalah seperti ini :

```
[In [30]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.26 (+/- 0.01)
```

**Gambar 3.33** Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

```
[# In [28]:
1 scoresvm = cross_val_score(clfsvm, df_train_att,
2                             df_train_label, cv=5) #sebagai data training
3 print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(),
4                                         scoresvm.std() * 2)) #sebagai data testing dan output
5 akurasi
```

Hasilnya adalah seperti ini :

```
[In [31]: scoresvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(), scoresvm.std() * 2))
Accuracy: 0.47 (+/- 0.03)
```

**Gambar 3.34** Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

```
[# In [29]: Pengamatan komponen informasi
1 max_features_opts = range(5, 50, 5) #max_features_opts sebagai
2     variabel untuk membuat range 5,50,5
3 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
4     sebagai variabel untuk membuat range 10,200,20
5 rf_params = np.empty((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=max_features
12             , n_estimators=n_estimators) #menampilkan variabel csf
13         scores = cross_val_score(clf, df.train_att,
14             df_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
18         rf_params[i,3] = scores.std() * 2 #index 3
19         i += 1 #dengan ketentuan i += 1
20         print("Max features: %d, num estimators: %d, accuracy:
21             %0.2f (+/- %0.2f)" % (max_features, n_estimators,
22             scores.mean(), scores.std() * 2))
```

```
17 #print hasil pengulangan yang sudah ditentukan
```

Hasilnya adalah seperti ini :

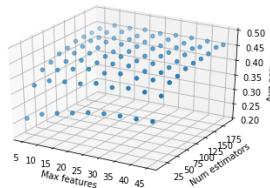
Max Features: 5, num estimators: 10, accuracy: 0.46 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 50, accuracy: 0.40 (+/- 0.03)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.03)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.01)	Max Features: 30, num estimators: 10, accuracy: 0.40 (+/- 0.02)
Max Features: 10, num estimators: 30, accuracy: 0.39 (+/- 0.03)	Max Features: 30, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 50, accuracy: 0.40 (+/- 0.03)	Max Features: 30, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 70, accuracy: 0.41 (+/- 0.03)	Max Features: 30, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 90, accuracy: 0.42 (+/- 0.03)	Max Features: 30, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 110, accuracy: 0.42 (+/- 0.03)	Max Features: 30, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 130, accuracy: 0.43 (+/- 0.03)	Max Features: 30, num estimators: 150, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 150, accuracy: 0.43 (+/- 0.03)	Max Features: 30, num estimators: 170, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 170, accuracy: 0.43 (+/- 0.03)	Max Features: 30, num estimators: 190, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 190, accuracy: 0.43 (+/- 0.03)	Max Features: 30, num estimators: 210, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 10, accuracy: 0.39 (+/- 0.02)	Max Features: 35, num estimators: 10, accuracy: 0.41 (+/- 0.03)
Max Features: 15, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 35, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 35, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 35, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 90, accuracy: 0.42 (+/- 0.03)	Max Features: 35, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 110, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 130, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 150, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 150, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 170, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 170, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 190, accuracy: 0.43 (+/- 0.03)
Max Features: 15, num estimators: 190, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 210, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 10, accuracy: 0.38 (+/- 0.02)	Max Features: 40, num estimators: 10, accuracy: 0.41 (+/- 0.02)
Max Features: 20, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 40, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 40, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 40, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 90, accuracy: 0.42 (+/- 0.03)	Max Features: 40, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 110, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 130, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 150, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 150, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 170, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 170, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 190, accuracy: 0.43 (+/- 0.03)
Max Features: 20, num estimators: 190, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 210, accuracy: 0.43 (+/- 0.03)
Max Features: 25, num estimators: 10, accuracy: 0.33 (+/- 0.02)	Max Features: 45, num estimators: 10, accuracy: 0.41 (+/- 0.02)
Max Features: 25, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 45, num estimators: 50, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 45, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 90, accuracy: 0.42 (+/- 0.03)	Max Features: 45, num estimators: 110, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 110, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 130, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 130, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 150, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 150, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 170, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 190, accuracy: 0.44 (+/- 0.03)
Max Features: 25, num estimators: 190, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 210, accuracy: 0.44 (+/- 0.03)

Gambar 3.35 Hasil Soal 8 - 1

- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```
1 # In [30]:
2
3 import matplotlib.pyplot as plt #import library matplotlib sebagai plt
4 from mpl_toolkits.mplot3d import Axes3D #import axes3D untuk menampilkan plot 3 dimensi
5 from matplotlib import cm #memanggil data cm yang sudah tersedia
6 fig = plt.figure() #hasil plot sebagai figure
7 fig.clf() #figure di ambil dari clf
8 ax = fig.gca(projection='3d') #ax sebagai projection 3d
9 x = rf_params[:,0] #x sebagai index 0
10 y = rf_params[:,1] #y sebagai index 1
11 z = rf_params[:,2] #z sebagai index 2
12 ax.scatter(x, y, z) #membuat plot scatter x y z
13 ax.set_zlim(0.2, 0.5) #set zlim dengan ketentuan yang ada
14 ax.set_xlabel('Max features') #memberi nama label x
15 ax.set_ylabel('Num estimators') #memberi nama label y
16 ax.set_zlabel('Avg accuracy') #memberi nama label z
17 plt.show() #print hasil plot yang sudah dibuat.
```

Hasilnya adalah seperti ini :



**Gambar 3.36** Hasil Soal 8 - 2

### 3.2.3 Penanganan Error

#### 1. ScreenShoot Error

```
[{ "err": "Error: Can't open file: /home/muhammadfaими/PycharmProjects/RandomForest/RandomForest/RandomForest.py", "file": "/home/muhammadfaими/PycharmProjects/RandomForest/RandomForest/RandomForest.py", "line": 1, "msg": "FileNotFoundError: [Errno 2] No such file or directory: '/home/muhammadfaими/PycharmProjects/RandomForest/RandomForest/RandomForest.py'", "stack": "FileNotFoundError: [Errno 2] No such file or directory: '/home/muhammadfaimi/PycharmProjects/RandomForest/RandomForest/RandomForest.py'" }]
```

**Gambar 3.37** FileNotFoundError

#### 2. Tuliskan Kode Error dan Jenis Error

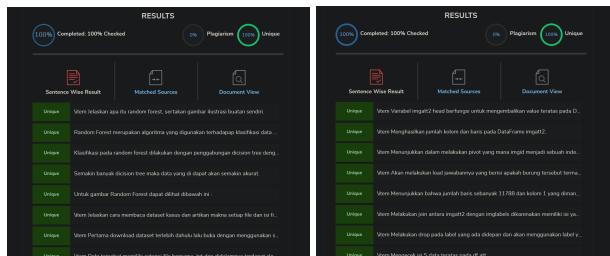
- FileNotFoundError

#### 3. Cara Penangan Error

- FileNotFoundError

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

### 3.2.4 Bukti Tidak Plagiat



**Gambar 3.38** Bukti Tidak Melakukan Plagiat Chapter 3

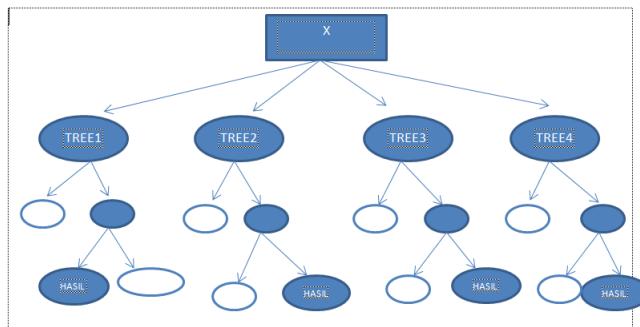
### 3.3 Muhammad Tomy 1174031

#### 3.3.1 Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest adalah konstruk data yang diterapkan pada machine learning yang mengembangkan sejumlah besar pohon keputusan acak yang menganalisis sekumpulan variabel. Jenis algoritma ini membantu meningkatkan cara teknologi menganalisis data yang kompleks. Juga merupakan algoritma machine learning yang fleksibel, mudah digunakan, bahkan tanpa penyetelan hyperparameter, dengan hasil yang baik. Ini juga merupakan salah satu algoritma yang paling banyak digunakan, karena kesederhanaan dan faktanya dapat digunakan untuk tugas klasifikasi dan regresi.

Dibawah ini merupakan salah satu ilustrasi penggunaan Random Forest.



**Gambar 3.39** Random Forest

2. Jelaskan cara membaca dataset khusus dan artikan makna setiap file dan isi field masing masing file. Dataset adalah kumpulan data. Paling umum satu data set sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, di mana setiap kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.

- Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
- Setelah itu, buat variabel baru "dataset" yang berisikan perintah untuk membaca file csv.
- Memanggil Librari Panda untuk membaca dataset
- Membuat variabel "Dataset" yang berisikan pdreadcsv untuk membaca dataset. Pada contoh ini menggunakan txt tapi tetap bisa membaca datasetnya, mengapa? Karena pada saat dijalankan librari panda secara otomatis akan mengubah data dalam bentuk text file ke format csv.

3. Jelaskan apa itu Cross Validation. Cross Validation adalah prosedur resampling yang digunakan untuk mengevaluasi model machine learning pada sampel data yang terbatas. Prosedur ini memiliki parameter tunggal yang disebut k yang mengacu pada jumlah grup tempat sampel data yang akan dibagi. Karena itu, prosedur ini sering disebut k-fold cross-validation. Proses penentuan apakah hasil numerik yang mengukur hubungan yang dihipotesiskan antar variabel, dapat diterima sebagai deskripsi data, dikenal sebagai Validationi. Umumnya, estimasi kesalahan untuk model dibuat setelah training, lebih dikenal sebagai evaluasi residu. Dalam proses ini, estimasi numerik dari perbedaan respons yang diprediksi dan yang asli dilakukan, juga disebut kesalahan training. Namun, ini hanya memberi kita gambaran tentang seberapa baik model kita pada data yang digunakan untuk melatihnya. Sekarang mungkin bahwa model tersebut kurang cocok atau overfitting data. Jadi, masalah dengan teknik evaluasi ini adalah bahwa itu tidak memberikan indikasi seberapa baik pelajar akan menggeneralisasi ke set data independen / tidak terlihat. Model ini dikenal sebagai Cross Validation.
4. Jelaskan apa arti score 44 % pada random forest, 27 % pada decision tree dan 29 % dari SVM. Itu merupakan persentase keakurasi prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasi atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decission Tree hasil prediksinya yaitu 27% dan pada SVM 29% .
5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri. Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.
  - 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 di plotting.

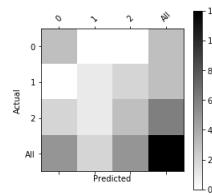
```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 11:54:37 2020
4
```

```

5 @author: ACER
6 """
7
8 import numpy as np
9 import matplotlib.pyplot as plt
10 import pandas as pd
11 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name='Actual')
12 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name='Predicted')
13 df_confusion = pd.crosstab(y_actu, y_pred)
14 df_confusion = pd.crosstab(y_actu, y_pred, rownames=['Actual'],
15                           colnames=['Predicted'], margins=True)
15 def plot_confusion_matrix(df_confusion, title='Confusion matrix',
16                           cmap=plt.cm.gray_r):
16     plt.matshow(df_confusion, cmap=cmap) # imshow
17     #plt.title(title)
18     plt.colorbar()
19     tick_marks = np.arange(len(df_confusion.columns))
20     plt.xticks(tick_marks, df_confusion.columns, rotation=45)
21     plt.yticks(tick_marks, df_confusion.index)
22     #plt.tight_layout()
23     plt.ylabel(df_confusion.index.name)
24     plt.xlabel(df_confusion.columns.name)
25 plot_confusion_matrix(df_confusion)
26 plt.show()

```

In [17]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/1.py', wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan')

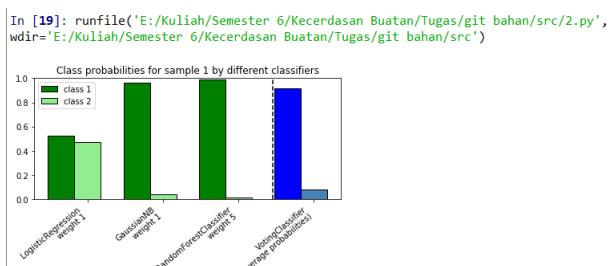


**Gambar 3.40** Confusion Matriks

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

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

- Untuk menggunakan Voting pada Random Forest dapat dilihat code berikut. Disini saya mengilustrasikan voting untuk berbagai macam algoritma terutama Random Forest.

**Gambar 3.41** Voting

### 3.3.2 Praktikum

#### 1. pandas

pada baris ke satu yaitu perintah mengimport library padas pada python atau anaconda kemudian di inisialisasikan menjadi karakter. selanjutnya ada sebuah array yang berisi a b c d. selanjutnya penggunaan array tipe series dan yang terakhir perintah print untuk menampilkan data pada karakter.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:06:12 2020
4
5 @author: ACER
6 """
7
8 import pandas as pd
9 data = np.array(['a', 'b', 'c', 'd'])
10 karakter = pd.Series(data)
11 print(karakter)
```

```

In [20]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/3.py',
      wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')
0    a
1    b
2    c
3    d
dtype: object
```

**Gambar 3.42** hasil

#### 2. numpy

Arti tiap baris codingan pada aplikasi sederhana numpy adalah sebagai berikut : pada baris ke satu yaitu mengimport numpy yang di inisialisasi menjadi np kemudian pada baris selanjutnya berisikan arange yang berarti membuat data yang berisi 12 dan ada reshape yang berfungsi merubah bentuk dari satu baris menjadi 2 baris data. Lalu yang terakhir ada perintah untuk print yaitu menampilkan data dari dika.

```
1 # -*- coding: utf-8 -*-
```

```

2 """
3 Created on Tue Mar 17 12:12:39 2020
4
5 @author: ACER
6 """
7
8 import numpy as np
9 tomy=np.arange(12).reshape(6,2)
10 print(tomy)

```

```

In [21]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/4.py',
      wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')
[[ 0  1]
 [ 2  3]
 [ 4  5]
 [ 6  7]
 [ 8  9]
 [10 11]]

```

**Gambar 3.43** hasil

### 3. matplotlib

Arti tiap baris aplikasi sederhana matplotlib pada baris ke satu yaitu memasukan library matplotlib.pyplot yang di definisikan menjadi plt kemudian plt.plot untuk menentukan grafik yang akan dibuat. lalu membuat variabel y dengan nama some numbers yang terakhir untuk menampilkan data pada sebuah grafik.

```

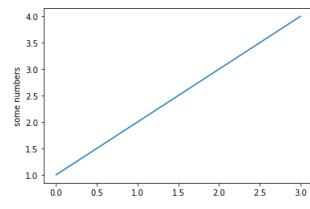
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:13:16 2020
4
5 @author: ACER
6 """
7
8 import matplotlib.pyplot as plt
9 plt.plot([1, 2, 3, 4])
10 plt.ylabel('some numbers')
11 plt.show()

```

```

In [22]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/5.py',
      wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')

```

**Gambar 3.44** hasil

### 4. Random Forest

Arti tiap baris hasil codingan random forest pada baris pertama random forest di import dari sklearn dengan ketentuan yaitu Nilai default untuk parameter yang mengontrol ukuran pohon (mis. Max\_depth, min\_samples\_leaf, dll.) Mengarah ke pohon yang tumbuh besar dan tidak di-unsun yang berpotensi sangat besar pada beberapa set data. Untuk mengurangi konsumsi memori, kompleksitas dan ukuran pohon harus dikontrol dengan menetapkan nilai parameter tersebut.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:15:21 2020
4
5 @author: ACER
6 """
7
8 from sklearn.ensemble import RandomForestClassifier
9 from sklearn.datasets import make_classification
10 X, y = make_classification(n_samples=1000, n_features=4,
11                           n_informative=2, n_redundant=0,
12                           random_state=0, shuffle=False)
13 clf = RandomForestClassifier(max_depth=2, random_state=0)
14 clf.fit(X, y)
15 print(clf.feature_importances_)
16 print(clf.predict([[0, 0, 0, 0]]))

In [27]: X, y = make_classification(n_samples=1000, n_features=4,
...:                           n_informative=2, n_redundant=0,
...:                           random_state=0, shuffle=False)
...: clf = RandomForestClassifier(max_depth=2, random_state=0)
...: clf.fit(X, y)
Out[27]:
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=2, max_features='auto',
                       max_leaf_nodes=None, max_samples=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)

In [28]: print(clf.feature_importances_)
[0.14205973 0.76664038 0.0282433 0.06305659]

In [29]: print(clf.predict([[0, 0, 0, 0]]))
[1]

```

**Gambar 3.45** hasil

## 5. Confusion Matrix

arti codingan pada hasil tiap codingan confusion matrix pada baris pertama codingan tersebut mendeskripsikan atau mengimport confusion matrix dari sklearn kemudian dibuat variabel y\_true untuk nilai target ground truth (benar). y\_pred untuk Taksiran target seperti yang dikembalikan oleh classifier. lalu menampilkan kedua variabel.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:17:15 2020
4
5 @author: ACER

```

```

6 """
7
8 from sklearn.metrics import confusion_matrix
9 y_true = [2, 0, 2, 2, 0, 1]
10 y_pred = [0, 0, 2, 2, 0, 2]
11 confusion_matrix(y_true, y_pred)

```

```

In [35]: from sklearn.metrics import confusion_matrix
...: y_true = [2, 0, 2, 2, 0, 1]
...: y_pred = [0, 0, 2, 2, 0, 2]
...: confusion_matrix(y_true, y_pred)
Out[35]:
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)

```

**Gambar 3.46** hasil

## 6. SVM dan Decision Tree

Seperti pengklasifikasi lainnya, DecisionTreeClassifier mengambil input dua array: array X, jarang atau padat, dengan ukuran n\_samples, n\_features memegang sampel pelatihan, dan array Y dari nilai integer, ukuran n\_samples, Atau, probabilitas setiap kelas dapat diprediksi. Seperti pengklasifikasi lainnya, SVC, NuSVC dan LinearSVC mengambil input dua array: array X ukuran n\_samples, n\_features memegang sampel pelatihan, dan array y label kelas (string atau bilangan bulat), ukuran n\_samples:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:17:49 2020
4
5 @author: ACER
6 """
7
8 from sklearn import tree
9 X = [[0, 0], [1, 1]]
10 Y = [0, 1]
11 clf = tree.DecisionTreeClassifier()
12 clf = clf.fit(X, Y)
13 clf.predict([[2., 2.]])
14
15 from sklearn import svm
16 X = [[0, 0], [1, 1]]
17 y = [0, 1]
18 clf = svm.SVC()
19 clf.fit(X, y)
20 clf.predict([[2., 2.]])

```

## 7. Cross Validation

digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut untuk lebih jelasnya dapat di lihat pada gambar pada codingan tersebut pada

```
In [40]: from sklearn import svm
...: X = [[0, 0], [1, 1]]
...: y = [0, 1]
...: clf = svm.SVC()
...: clf.fit(X, y)
...: clf.predict([[2., 2.]])
Out[40]: array([1])

In [41]: from sklearn import tree
...: X = [[0, 0], [1, 1]]
...: Y = [0, 1]
...: clf = tree.DecisionTreeClassifier()
...: clf = clf.fit(X, Y)
...: clf.predict([[2., 2.]])
Out[41]: array([1])
```

**Gambar 3.47** hasil

baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel lontong setelah hal tersebut dilakukan kemudian data tersebut di eksekusi. berikut merupakan hasil dari code tersebut dapat dilihat pada gambar.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:18:20 2020
4
5 @author: ACER
6 """
7
8 from sklearn.model_selection import cross_val_score
9 scores = cross_val_score(lontong, pecel_att, pecel_pass, cv=5)
10 # show average score and +/- two standard deviations away
11 #(covering 95% of scores)
12 print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std()
13 () * 2))
```

```
In [61]: runfile('D:/SEMESTER 6/wert/9.py', wdir='D:/SEMESTER 6/wert')
Accuracy: 0.59 (+/- 0.05)
```

**Gambar 3.48** hasil

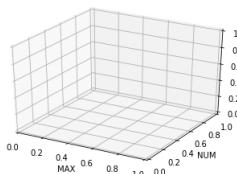
8. program pengamatan arti dari hasil program pengamatan. perogram pengamatan dapat mengamati dari 3 aspek diatas yaitu svm, random, dan decision tree. Yang memiliki variabel X Y Z dan di tampilkan dalam bentuk grafik.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:19:01 2020
4
5 @author: ACER
6 """
7
8 import matplotlib.pyplot as plt
9 from mpl_toolkits.mplot3d import Axes3D
10 from matplotlib import cm
11 fig = plt.figure()
```

```

12 fig . clf ()
13 ax = fig . gca ( projection = ' 3d ' )
14 plt . xlabel ( ' MAX ' )
15 plt . ylabel ( ' NUM ' )
16 plt . zlabel ( ' AVG ' )
17 ax . scatter ( x , y , z )
18 ax . set_zlim ( 0.2 , 0.5 )
19 ax . set_xlabel ( ' Max features ' )
20 ax . set_ylabel ( ' Num estimator ' )
21 ax . set_zlabel ( ' Avg accuracy ' )
22 plt . show ()

```



**Gambar 3.49** hasil

### 3.3.3 Penanganan Error

Screenshot error

1. Untuk gambar screenshot error

```

File "<ipython-input-60-d14a3944647ax>", line 1, in <module>
    runfile('D:/SEMESTER 6/wert/1/5.py', wdir='D:/SEMESTER 6/wert/1')

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 827, in runfile
    execfile(filename, namespace)

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 110, in execfile
    exec(compile(f.read(), filename, 'exec'), namespace)

File "D:/SEMESTER 6/wert/1/5.py", line 10
    plt.ylabel(some numbers)
                           ^
SyntaxError: invalid syntax

```

**Gambar 3.50** hasil

Code Errornya

```

import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel(some numbers)
plt.show()

```

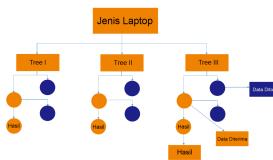
pada kode ylabel memiliki nama atau isi some numbers tetapi pada tipe data tertentu harus di awali dan diakhiri dengan tanda petik 2 atau 1. Pada kodingnya hanya kurang tanda petik 1.

## 3.4 1174005 - Oniwaldus Bere Mali

### 3.4.1 Soal Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest merupakan algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakukn training terhadap sempel data yang dimiliki. Semakin banyak dicision tree maka data yang di dapat akan semakin akurat. Untuk gambar Random Forest dapat dilihat dibawah ini :



Gambar 3.51 Random Forest.

2. Jelaskan 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. Jelaskan apa itu cross validation.

Cross Validation merupakan 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.

4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh 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 decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision 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.

5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

1 from matplotlib import cm #memanggil data cm yang sudah tersedia
2 fig = plt.figure() #hasil plot sebagai figure
3 fig.clf() #figure di ambil dari clf
4 ax = fig.gca(projection='3d') #ax sebagai projection 3d
5 x = rf_params[:,0] #x sebagai index 0

```

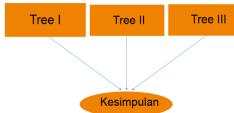
Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data positif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data positif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat dibawah ini :



**Gambar 3.52** Confusion Matriks.

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri

Voting pada random forest ialah sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk lebih jelasnya saya akan memberikan sebuah contoh bagaimana voting bekerja :



**Gambar 3.53** Decision Tree.

Dimana ditunjukkan pada gambar diatas terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan diadakan voting untuk menentukan sebuah laptop. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan laptop asus, yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah laptop asus. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menyatakan laptop asus sedangkan pada tree 3 menyatakan laptop HP, maka kesimpulan yang diambil adalah laptop asus dikarenakan hasil voting terbanyak adalah laptop asus.

### 3.4.2 Praktek Program

1. Buat aplikasi sederhana menggunakan pandas dan jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas).

```

1 # In [44]: Soal1
2
3 import pandas as oni #melakukan import pada library pandas
4          sebagai oni
5 laptop = {"Nama Laptop" : ['Asus', 'HP', 'Lenovo', 'Samsung']} #
6          membuat varibel yang bernama laptop , dan mengisi dataframe
7          nama2 laptop
  
```

Kode di atas digunakan untuk mengimpor atau mengirim library pandas sebagai fahmi, Hasilnya adalah sebagai berikut :

	Nama Laptop
0	Fahmi Punya Laptop Asus
1	Fahmi Punya Laptop HP
2	Fahmi Punya Laptop Lenovo
3	Fahmi Punya Laptop Samsung

**Gambar 3.54** Hasil Soal 1.

2. buat aplikasi sederhana menggunakan numpy dan jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```

1 import numpy as oni
2 matrix_x = oni.eye(10) #membuat matrix dengan numpy dengan
   menggunakan fungsi eye
3 matrix_x #deklrasikan matrix_x yang telah dibuat
4
5 print (matrix_x) #print matrix_x yang telah dibuat dengan 10x10

```

Fungsi eye disini berguna untuk memanggil matrix identitas dengan jumlah colom dan baris sesuai yang ditentukan. Disini saya telah menentukan 10 colom dan baris maka dari itu hasilnya akan memunculkan matrix identitas 10x10, Hasilnya adalah sebagai berikut :

```

[[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 3.55** Hasil Soal 2.

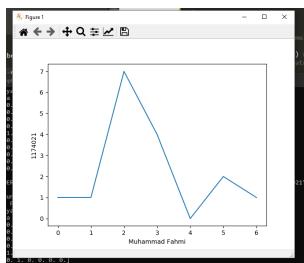
3. buat aplikasi sederhana menggunakan matplotlib dan jelaskan arti dari setiap baris kode(harus beda dengan teman sekelas)

```

1
2 oni.plot([1,1,7,4,0,0,5]) #memberikan nilai plot atau grafik pada
   fahmi
3 oni.xlabel('Oniwaldus Bere Mali') #memberikan label pada x
4 oni.ylabel('1174021') #memberikan label pada y
5 oni.show() #print hasil plot berbentuk grafik

```

Jalankan source code diatas dengan spyder atau anaconda sehingga hasilnya akan seperti berikut :

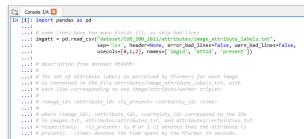
**Gambar 3.56** Hasil Soal 3.

4. jalankan program klasifikasi Random Forest pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- 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.

```
1 # In [1]: RANDOM FOREST
2
3 import pandas as pd #import library pandas sebagai as
4
5 imgatt = pd.read_csv("D:/KULIAH/SEMESTER 6/Kecerdasan Buatan/
   Chapter 3/src/1174005/dataset/CUB_200_2011/attributes/
   image_attribute_labels.txt",
   sep='\s+', header=None, error_bad_lines=
   False, warn_bad_lines=False,
   usecols=[0,1,2], names=['imgid', 'attid',
   'present']) #library imgatt sebagai membaca file csv dari
dataset, dengan ketentuan yang ada.
```

Hasilnya adalah seperti ini :

**Gambar 3.57** Hasil Soal 4 - 1

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

```
1 # In [2]:
2
3 imgatt.head() #menampilkan data yang di baca tadi tetapi hanya
   data paling atas
```

Hasilnya adalah seperti ini :

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

**Gambar 3.58** Hasil Soal 4 - 2

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

```
1 # In [3]:
2
3 imgatt.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

```
In [3]: imgatt.shape
Out[3]: (3677856, 3)
```

**Gambar 3.59** Hasil Soal 4 - 3

- Variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya.

```
1 # In [4]:
2
3 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values=
   'present') #membuat variabel baru dari fungsi imgatt ,
   dengan mengganti index dan kolom (kebalikan)
```

Hasilnya adalah seperti ini :

```
In [4]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
```

**Gambar 3.60** Hasil Soal 4 - 4

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

```
1 # In [5]:
2
3 imgatt2.head() #menampilkan data yang di baca tadi tetapi
   hanya data paling atas
```

Hasilnya adalah seperti ini :

```
[In [5]: imgatt2.head()
Out[5]:
   imgid  1  2  3  4  5  6  7  ... 386 387 388 389 390 391 392
   0      0  0  0  0  0  1  0  ... 0  0  0  1  0  0  0
   1      0  0  0  0  0  0  0  ... 0  0  0  0  0  0  0
   2      0  0  0  0  0  0  0  ... 0  0  0  1  0  0  0
   3      0  0  0  0  0  0  0  ... 0  0  0  0  0  0  0
   4      0  0  0  0  0  0  1  ... 0  0  0  0  0  0  0
   5      0  0  0  0  0  0  0  ... 0  0  0  0  0  0  0
[5 rows x 312 columns]
```

**Gambar 3.61** Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 # In [6]:
2
3 imgatt2.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

In [6]:	imgatt2.shape
Out[6]:	(11788, 312)

**Gambar 3.62** Hasil Soal 4 - 6

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

```
1 # In [7]:
2
3 imglabels = pd.read_csv("D:/KULIAH/SEMESTER 6/Kecerdasan
   Buatan/Chapter 3/src/1174005/dataset/CUB_200_2011/
   image_class_labels.txt",
   sep=' ', header=None, names=['imgid',
   'label']) #baca data csv dengan ketentuan yang ada
5
6 imglabels = imglabels.set_index('imgid') #variabel imglabels
   sebagai set index (imgid)
```

Hasilnya adalah seperti ini :

```
[In [8]: imglabels = pd.read_csv("D:/KULIAH/SEMESTER 6/Kecerdasan
   Buatan/Chapter 3/src/1174005/dataset/CUB_200_2011/
   image_class_labels.txt",
   sep=' ', header=None, names=['imgid',
   'label'])
   ...
   imglabels = imglabels.set_index('imgid')
   ...
   # A description from dataset README:
   ...
   # The ground truth class labels (bird species labels) for each image are contained
   # in the file image_class_labels.txt, with each line corresponding to one image;
   # the first column is the image id and the second column is the class label;
   # there, image_id and classes_id correspond to the IDs in images.txt and classes.txt,
   # respectively.
[In [8]: imglabels.head()
Out[8]:
   imgid  label
   0      1
   1      1
   2      1
   3      1
   4      1
```

**Gambar 3.63** Hasil Soal 4 - 7

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

```
1 # In [8]:
2
3 imglabels.head() #menampilkan data yang di baca tadi tetapi
   hanya data paling atas
```

Hasilnya adalah seperti ini :

```
In [9]: imglabels.head()
Out[9]:
   label
  imgid
  1      1
  2      1
  3      1
  4      1
  5      1
```

**Gambar 3.64** Hasil Soal 4 - 8

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

```
1 # In [9]:
2
3 imglabels.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

```
In [10]: imglabels.shape
Out[10]: (11788, 1)
```

**Gambar 3.65** Hasil Soal 4 - 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 # In [10]:
2
3 df = imgatt2.join(imglabels) #varibel df sebagai fungsi join
    dari data imgatt2 ke varibel imglabels
4 df = df.sample(frac=1) #varibel df sebagai sample dengan
    ketentuan frac=1
```

Hasilnya adalah seperti ini :

```
In [11]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
```

**Gambar 3.66** Hasil Soal 4 - 10

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

```
1 # In [11]:
2
3 df_att = df.iloc[:, :312] #membuat kolom dengan ketentuan 312
4 df_label = df.iloc[:, 312:] #membuat kolom dengan ketentuan
    312
```

Hasilnya adalah seperti ini :

```
In [12]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

**Gambar 3.67** Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```
1 # In [12]:
2
3 df_att.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas
```

Hasilnya adalah seperti ini :

	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312
imgid	1	0	0	0	0	0	1	...	0	0	0	0	0	0	1
2449	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
988	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1
7084	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9834	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1

[5 rows x 312 columns]

**Gambar 3.68** Hasil Soal 4 - 12

- Mengecek isi data teratas dari df label.

```
1 # In [13]:
2
3 df_label.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas
```

Hasilnya adalah seperti ini :

	label
imgid	43
2449	18
988	128
7472	121
7084	168

**Gambar 3.69** Hasil Soal 4 - 13

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

```
1 # In [14]:
2
3 df_train_att = df_att[:8000] #akan membagi 8000 row pertama menjadi data training dan sisanya adalah data testing
4 df_train_label = df_label[:8000] #akan membagi 8000 row pertama menjadi data training dan sisanya adalah data testing
```

```

5 df_test_att = df_att[8000:] #kebalikan dari akan membagi 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
6 df_test_label = df_label[8000:] #kebalikan dari akan membagi
   8000 row pertama menjadi data training dan sisanya adalah
   data testing
7
8 df_train_label = df_train_label['label'] #menampilkan data
   training
9 df_test_label = df_test_label['label'] #menampilkan data
   testing

```

Hasilnya adalah seperti ini :

```

In [15]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...:
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']

```

**Gambar 3.70** Hasil Soal 4 - 14

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

```

1 # In [15]:
2
3 from sklearn.ensemble import RandomForestClassifier #import
   randomforestclassifier
4 clf = RandomForestClassifier(max_features=50, random_state=0,
   n_estimators=100) #clf sebagai variabel untuk klasifikasi
   random forest

```

Hasilnya adalah seperti ini :

```

In [16]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)

```

**Gambar 3.71** Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 # In [16]:
2
3 clf.fit(df_train_att, df_train_label) #variabel clf untuk fit
   yaitu training

```

Hasilnya adalah seperti ini :

```

In [17]: clf.fit(df_train_att, df_train_label)
Out[17]:
RandomForestClassifier(n_estimators=100, criterion='gini', max_depth=None, max_features=50,
   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,
   oob_score=False, random_state=0, verbose=0,
   warm_start=False)

```

**Gambar 3.72** Hasil Soal 4 - 16

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

```

1 # In [17]:
2
3 print(clf.predict(df_train_att.head())) #print clf yang di
    prediksi dari training tetapi hanya menampilkan data
    paling atas

```

Hasilnya adalah seperti ini :

```

In [18]: print(clf.predict(df_train_att.head()))
[ 43 18 128 121 169]

```

**Gambar 3.73** Hasil Soal 4 - 17

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

```

1 # In [18]:
2
3 clf.score(df_test_att, df_test_label) #print clf sebagai
    testing yang sudah di training tadi

```

Hasilnya adalah seperti ini :

```

In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.4390179514255544

```

**Gambar 3.74** Hasil Soal 4 - 18

- Jalankan program confusion matrix pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Melakukan import confusion matrix pada library sklearn matriks.

```

1 # In [19]: Confusion Matrix
2
3 from sklearn.metrics import confusion_matrix #import Confusion
    Matrix
4 pred_labels = clf.predict(df_test_att) #sebagai data testing
5 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
    variabel data label

```

Hasilnya adalah seperti ini :

```

In [20]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)

```

**Gambar 3.75** Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```

1 # In [20]:
2
3 cm #menampilkan data label berbentuk array

```

Hasilnya adalah seperti ini :

```

In [21]: cm
Out[21]:
array([[ 8.,  1.,  1., ...,  0.,  1.,  0.],
       [ 0., 11.,  0., ...,  0.,  0.,  0.],
       [ 0.,  1.,  6., ...,  0.,  0.,  0.],
       ...,
       [ 0.,  0.,  0., ...,  1.,  0.,  0.],
       [ 0.,  0.,  0., ...,  0.,  3.,  0.],
       [ 0.,  0.,  0., ...,  0.,  0., 16.]], dtype=int64)

```

**Gambar 3.76** Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```

1
2 # In [21]:
3
4 import matplotlib.pyplot as plt #import library matplotlib
      sebagai plt
5 import itertools #import library itertools
6 def plot_confusion_matrix(cm, classes,
7                           normalize=False,
8                           title='Confusion matrix',
9                           cmap=plt.cm.Blues): #membuat fungsi
    dengan ketentuan data yang ada pada cm
10
11 if normalize:
12     cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13     print("Normalized confusion matrix") #jika normalisasi
        sebagai ketentuan yang ada maka print normalized
        confusion matrix
14 else:
15     print('Confusion matrix, without normalization') #jika
        tidak memenuhi kondisi if maka pring else
16
17 print(cm) #print data cm
18
19 plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
        sebagai fungsi untuk membuat plot
20 plt.title(title) #membuat title pada plot
# plt.colorbar()
21 tick_marks = np.arange(len(classes)) #membuat marks pada
        plot
22 plt.xticks(tick_marks, classes, rotation=90) #membuat
        ticks pada x
23 plt.yticks(tick_marks, classes) #membuat ticks pada y
24
25 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
26 thresh = cm.max() / 2. #variabel thresh menambil data max
        pada cm kemudian dibagi 2
27
28 plt.tight_layout() #mengatur layout pada plot
29 plt.ylabel('True label') #memberi nama label pada sumbu y

```

```

31 plt.xlabel('Predicted label') #memberi nama label pada
32 sumbu x
33
34 # In [22]:
35
36 birds = pd.read_csv("data/CUB_200_2011/classes.txt",
37 sep='\s+', header=None, usecols=[1], names
38 =['birdname']) #membaca csv dengan ketentuan nama birdname
39 birds = birds[['birdname']] #nama birds dengan ketentuan
40 birdname
41 birds #menampilkan data birds
42
43
44 # In [23]:
45
46 import numpy as np #import library numpy sebagai np
47 np.set_printoptions(precision=2) #np sebagai variabel yang
48 membuat set precision=2
49 plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure
50 dengan ketentuan sizw 60,60 dan dpi 300
51 plot_confusion_matrix(cm, classes=birds, normalize=True) #data
52 cm dan clas birds dibuat sebagai plot

```

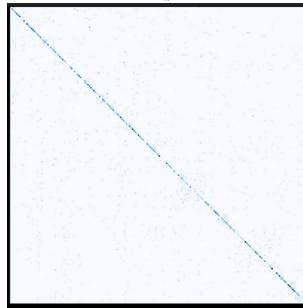
Hasilnya adalah seperti dibawah ini :

```

In [22]: import matplotlib.pyplot as plt
... import itertools;
... def plot_confusion_matrix(cm,
... classes,
... normalize=False,
... title='Confusion matrix',
... cmap=plt.cm.Blues):
...     """
... This function prints and plots the confusion matrix.
... Normalization can be applied by setting `normalize=True`.
... """
... if not isinstance(cm, np.ndarray):
...     cm = np.array(cm).astype('float')
...     cm = cm / cm.sum(axis=1)[:, np.newaxis]
... else:
...     cm = cm.astype('float')
...     cm = cm / cm.sum(axis=1)[:, np.newaxis]
...
... print("Confusion matrix, without normalization")
...
... print(cm)
...
... plt.imshow(cm, interpolation='nearest', cmap=cmap)
... plt.title(title)
... plt.colorbar()
... tick_marks = np.arange(len(classes))
... plt.xticks(tick_marks, classes, rotation=45)
... plt.yticks(tick_marks, classes, rotation=45)
...
... thresh = cm.max() / 2.
...
... for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
...     if cm[i, j] > thresh:
...         color = 'darkred'
...     else:
...         color = 'black'
...     plt.text(j, i, cm[i, j],
...             horizontalalignment="center",
...             verticalalignment="center",
...             color=color)
...
... plt.tight_layout()
... plt.ylabel('True label')
... plt.xlabel('Predicted label')
...
... plt.show()

In [23]: birds = pd.read_csv("data/CUB_200_2011/classes.txt",
... sep='\s+', header=None, usecols=[1], names=['birdname'])
... birds
Out[23]:
0 001.Black_Footed_Albatross
1 002.Laysan_Albatross
2 003.Gull_billed_Tern
3 004.Gull_billed_Tern
4 005.House_Sparrow
5 006.Myna_Sparrow
6 007.Mallard_Duck
7 008.Winter_Sparrow
8 009.Commont_Gull
9 010.Commont_Gull
Name: birdname, Length: 200, dtype: object

```



**Gambar 3.77** Menampilkan hasil plot

6. Jalankan program klasifikasi SVM dan Decission Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Menunjukkan klarifikasi dengan decision tree menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 # In [24]: Mencoba dengan metode Decision Tree dan SVM
2
3 from sklearn import tree #import library tree
4 clftree = tree.DecisionTreeClassifier() #clftree sebagai
5     variabel untuk decision tree
6 clftree.fit(df_train_att, df_train_label) #sebagai data
    training

```

Hasilnya adalah seperti ini :

```

In [27]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[27]: 0.26135163674762407

```

**Gambar 3.78** Hasil Soal 6 - 1

- Menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 # In [25]:
2
3 from sklearn import svm #import library svm
4 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
5     fungsi SVC
6 clfsvm.fit(df_train_att, df_train_label) #sebagai data
    training

```

Hasilnya adalah seperti ini :

```

In [28]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
Out[28]: 0.47729672650475186

```

**Gambar 3.79** Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

```

1 # In [26]: Pengecekan Cross Validation
2
3 from sklearn.model_selection import cross_val_score #import
4     cross_val_score
5 scores = cross_val_score(clf, df_train_att, df_train_label, cv
6     =5) #variabel scores sebagai variabel prediksi dari data
    training

```

Hasilnya adalah seperti ini :

```
[In [20]:] from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
...: print("Accuracy: %0.4f (+/- %0.03)
```

**Gambar 3.80** Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

```
1 # In [27]:
2
3
4 scorestree = cross_val_score(clftree, df_train_att,
...: df_train_label, cv=5) #sebagai prediksi menggunakan scores
...: dan metode tree
```

Hasilnya adalah seperti ini :

```
[In [20]:] scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
...: Accuracy: 0.26 (+/- 0.03)
```

**Gambar 3.81** Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

```
1
2 # In [28]:
3
4 scoresvm = cross_val_score(clfsvm, df_train_att,
...: df_train_label, cv=5) #sebagai data training
```

Hasilnya adalah seperti ini :

```
[In [21]:] scoresvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(), scoresvm.std() * 2))
...: Accuracy: 0.47 (+/- 0.03)
```

**Gambar 3.82** Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

```
1
2 # In [29]: Pengamatan komponen informasi
3
4 max_features_opts = range(5, 50, 5) #max_features_opts sebagai
...: variabel untuk membuat range 5,50,5
5 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
...: sebagai variabel untuk membuat range 10,200,20
```

```

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

```

Hasilnya adalah seperti ini :

Max Features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 20, accuracy: 0.28 (+/- 0.02)	Max Features: 25, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 30, accuracy: 0.30 (+/- 0.02)	Max Features: 25, num estimators: 200, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 40, accuracy: 0.30 (+/- 0.03)	Max Features: 30, num estimators: 50, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 30, num estimators: 100, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.03)	Max Features: 30, num estimators: 150, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 90, accuracy: 0.41 (+/- 0.03)	Max Features: 30, num estimators: 200, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max Features: 35, num estimators: 50, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 130, accuracy: 0.44 (+/- 0.03)	Max Features: 35, num estimators: 100, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max Features: 35, num estimators: 150, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)	Max Features: 35, num estimators: 200, accuracy: 0.49 (+/- 0.03)
Max Features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 50, accuracy: 0.45 (+/- 0.04)
Max Features: 5, num estimators: 210, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 100, accuracy: 0.45 (+/- 0.04)
Max Features: 5, num estimators: 230, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 150, accuracy: 0.45 (+/- 0.04)
Max Features: 5, num estimators: 250, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 200, accuracy: 0.45 (+/- 0.04)
Max Features: 10, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.45 (+/- 0.04)
Max Features: 10, num estimators: 20, accuracy: 0.28 (+/- 0.02)	Max Features: 38, num estimators: 250, accuracy: 0.45 (+/- 0.04)
Max Features: 10, num estimators: 30, accuracy: 0.30 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 40, accuracy: 0.30 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 50, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 130, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 150, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 170, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 190, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 210, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 230, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 10, num estimators: 250, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 20, accuracy: 0.28 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 30, accuracy: 0.30 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 40, accuracy: 0.30 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 90, accuracy: 0.43 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 150, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 170, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 190, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 210, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 230, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 15, num estimators: 250, accuracy: 0.44 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 20, accuracy: 0.28 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 30, accuracy: 0.30 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 40, accuracy: 0.30 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 50, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 110, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 130, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 170, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 190, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 210, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 230, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 20, num estimators: 250, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 20, accuracy: 0.28 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 30, accuracy: 0.30 (+/- 0.02)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 40, accuracy: 0.30 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 50, accuracy: 0.43 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 70, accuracy: 0.43 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 110, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 130, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 170, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 190, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 210, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 230, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)
Max Features: 25, num estimators: 250, accuracy: 0.44 (+/- 0.03)	Max Features: 38, num estimators: 180, accuracy: 0.46 (+/- 0.04)

Gambar 3.83 Hasil Soal 8 - 1

- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```

1
2 # In [30]:
3
4 import matplotlib.pyplot as plt #import library matplotlib
5         sebagai plt
6 from mpl_toolkits.mplot3d import Axes3D #import axes3D untuk
7         menampilkan plot 3 dimensi
8 from matplotlib import cm #memanggil data cm yang sudah
9         tersedia
10 fig = plt.figure() #hasil plot sebagai figure
11 fig.clf() #figure di ambil dari clf
12 ax = fig.gca(projection='3d') #ax sebagai projection 3d
13 x = rf_params[:,0] #x sebagai index 0
14 y = rf_params[:,1] #y sebagai index 1

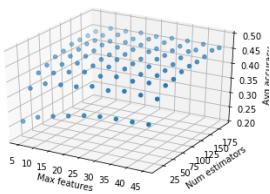
```

```

12 z = rf_params[:, 2] # z sebagai index 2
13 ax.scatter(x, y, z) #membuat plot scatter x y z
14 ax.set_zlim(0.2, 0.5) #set zlim dengan ketentuan yang ada
15 ax.set_xlabel('Max features') #memberi nama label x
16 ax.set_ylabel('Num estimators') #memberi nama label y
17 ax.set_zlabel('Avg accuracy') #memberi nama label z

```

Hasilnya adalah seperti ini :



**Gambar 3.84** Hasil Soal 8 - 2

### 3.4.3 Penanganan Error

#### 1. ScreenShoot Error

```

[2]: reader = pd.read_csv('diabetes.csv', header=0)
      reader['Outcome'].value_counts()
      reader['Outcome'].value_counts().plot(kind='bar')
      reader['Outcome'].value_counts().plot(kind='bar')

[3]: reader = pd.read_csv('diabetes.csv', header=0)
      reader['Outcome'].value_counts()
      reader['Outcome'].value_counts().plot(kind='bar')
      reader['Outcome'].value_counts().plot(kind='bar')

[4]: reader['pcut'] = not defined

```

**Gambar 3.85** FileNotFoundError

#### 2. Tuliskan Kode Error dan Jenis Error

- FileNotFoundError

#### 3. Cara Penangan Error

- FileNotFoundError

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

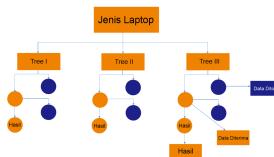
## 3.5 1174027 - Harun Ar - Rasyid

### 3.5.1 Teori

#### 1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest merupakan algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan decision tree dengan melakuakn training terhadap sempel

data yang dimiliki. Semakin banyak decision tree maka data yang di dapat akan semakin akurat. Untuk gambar Random Forest dapat dilihat dibawah ini :



**Gambar 3.86** Random Forest.

2. Jelaskan 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. Jelaskan apa itu cross validation.

Cross Validation merupakan 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 mengujinya dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh 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 decision

tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision 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.

5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

1 fig = plt.figure() #hasil plot sebagai figure
2 fig.clf() #figure di ambil dari clf
3 ax = fig.gca(projection='3d') #ax sebagai projection 3d
4 x = rf_params[:,0] #x sebagai index 0
5 y = rf_params[:,1] #y sebagai index 1

```

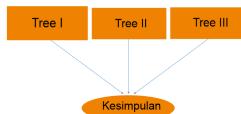
Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data positif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data positif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat dibawah ini :



**Gambar 3.87** Confusion Matriks.

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri

Voting pada random forest ialah sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk lebih jelasnya saya akan memberikan sebuah contoh bagaimana voting bekerja :



**Gambar 3.88** Decision Tree.

Dimana ditunjukkan pada gambar diatas terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan diadakan voting untuk menentukan sebuah laptop. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan laptop asus, yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah laptop asus. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menyatakan laptop asus sedangkan pada tree 3 menyatakan laptop HP, maka kesimpulan yang diambil adalah laptop asus dikarenakan hasil voting terbanyak adalah laptop asus.

### 3.5.2 Praktek

1. Buat aplikasi sederhana menggunakan pandas dan jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas).

```

1 import pandas as pd #digunakan untuk import library pandas
2 harun = pd.read_csv('dataku.csv',sep=';') #digunakan untuk
   memanggil file csv dan dipisahkan dengan ;
3 len(harun) #untuk mengetahui jumlah data
4 print(harun.nama) #digunakan untuk mengetahui kolom nama dari
   varivel harun
  
```

Kode di atas digunakan untuk mengimpor atau mengirim library pandas sebagai pd, Hasilnya adalah sebagai berikut :

0	Harun Ar - Rasyid
1	Evietania charis sujadi
2	Nico Sembiring
3	Magdalena
4	Luigi Intan
5	Nadya Rahma
6	Habib Abdul Rasyid
7	Etika Khusnul Laeli
8	Tomy Nur Maulidi

**Gambar 3.89** Hasil Soal 1.

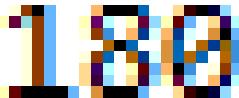
2. buat aplikasi sederhana menggunakan numpy dan jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```

1 import numpy as np #digunakan untuk import library numpy
2 totumur = np.sum(harun.umur) # menggunakan fungsi sum bawaan dari
   numpy
3 print(totumur) #menampilkan isi variabel umur

```

Fungsi sum disini berguna untuk menjumlahkan data. Hasilnya adalah sebagai berikut :



**Gambar 3.90** Hasil Soal 2.

3. buat aplikasi sederhana menggunakan matplotlib dan jelaskan arti dari setiap baris kode(harus beda dengan teman sekelas)

```

1 import matplotlib.pyplot as mt #import matplotlib
2 mt.plot([10,20,30,40,50],[1,2,3,4,5]) #digunakan untuk membuat
   fungsi graph
3 mt.xlabel("Kecepatan Mobil") #membuat label untuk garis x
4 mt.ylabel("Gigi") # membuat label untuk y
5 mt.title("Perubahan Kecepatan Mobil") #membuat judul graph
6 mt.show() # menampilkan graph

```

Jalankan source code diatas dengan spyder atau anaconda sehingga hasilnya akan seperti berikut :



**Gambar 3.91** Hasil Soal 3.

4. jalankan program klasifikasi Random Forest pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- 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.

```
1 import pandas as pd #import library pandas sebagai as
2
3 imgatt = pd.read_csv("data/CUB_200_2011/attributes/
4     image_attribute_labels.txt",
5     sep='\s+', header=None, error_bad_lines=
6     False, warn_bad_lines=False,
7     usecols=[0,1,2], names=['imgid', 'attid',
8     'present']) #library imgatt sebagai membaca file csv dari
9     dataset, dengan ketentuan yang ada.
```

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

```
1 imgatt.head() #menampilkan data yang di baca tadi tetapi hanya
2     data paling atas
```

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

```
1 imgatt.shape #menampilkan jumlah data , kolom
```

- Variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya.

```
1 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values=
2     'present') #membuat variabel baru dari fungsi imgatt ,
3     dengan mengganti index dan kolom (kebalikan)
```

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

```
1 imgatt2.head() #menampilkan data yang di baca tadi tetapi
2     hanya data paling atas
```

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 imgatt2.shape #menampilkan jumlah data , kolom
```

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

```
1 imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.
2     txt",
3     sep=' ', header=None, names=['imgid',
4     'label']) #baca data csv dengan ketentuan yang ada
5
6 imglabels = imglabels.set_index('imgid') #variabel imglabels
7     sebagai set index (imgid)
```

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

```
1 imglabels.head() #menampilkan data yang di baca tadi tetapi
    hanya data paling atas
```

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

```
1 imglabels.shape #menampilkan jumlah data , kolom
```

- 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 df = imgatt2.join(imglabels) #variabel df sebagai fungsi join
    dari data imgatt2 ke variabel imglabels
2 df = df.sample(frac=1) #variabel df sebagai sample dengan
    ketentuan frac=1
```

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

```
1 df_att = df.iloc[:, :312] #membuat kolom dengan ketentuan 312
2 df_label = df.iloc[:, 312:] #membuat kolom dengan ketentuan
    312
```

- Mengecek isi 5 data teratas pada df att.

```
1 df_att.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas
```

- Mengecek isi data teratas dari df label.

```
1 df_label.head() #menampilkan data yang di baca tadi tetapi
    hanya data paling atas
```

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

```
1 df_train_att = df_att[:8000] #akan membagi 8000 row pertama
    menjadi data training dan sisanya adalah data testing
2 df_train_label = df_label[:8000] #akan membagi 8000 row
    pertama menjadi data training dan sisanya adalah data
    testing
3 df_test_att = df_att[8000:] #kebalikan dari akan membagi 8000
    row pertama menjadi data training dan sisanya adalah data
    testing
4 df_test_label = df_label[8000:] #kebalikan dari akan membagi
    8000 row pertama menjadi data training dan sisanya adalah
    data testing
5
6 df_train_label = df_train_label['label'] #menampilkan data
    training
7 df_test_label = df_test_label['label'] #menampilkan data
    testing
```

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

```

1 from sklearn.ensemble import RandomForestClassifier #import
      randomforestclassifier
2 clf = RandomForestClassifier(max_features=50, random_state=0,
      n_estimators=100) #clf sebagai variabel untuk klafifikasi
      random forest

```

- Menunjukkan hasil prediksi dari Random Forest.

```

1 clf.fit(df_train_att, df_train_label) #variabel clf untuk fit
      yaitu training

```

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

```

1 print(clf.predict(df_train_att.head())) #print clf yang di
      prediksi dari training tetapi hanya menampilkan data
      paling atas

```

5. Jalankan program confusion matrix pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Melakukan import confusion matrix pada library sklearn matriks.

```

1 from sklearn.metrics import confusion_matrix #import Confusion
      Matrix
2 pred_labels = clf.predict(df_test_att) #sebagai data testing
3 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
      variabel data label

```

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```

1 cm #menampilkan data label berbentuk array

```

- Membuat dan menampilkan hasil plot.

```

1 import matplotlib.pyplot as plt #import library matplotlib
      sebagai plt
2 import itertools #import library itertools
3 def plot_confusion_matrix(cm, classes,
      normalize=False,
      title='Confusion matrix',
      cmap=plt.cm.Blues): #membuat fungsi
      dengan ketentuan data yang ada pada cm
4
5     if normalize:
6         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
7     ]
8         print("Normalized confusion matrix") #jika normalisasi
      sebagai ketentuan yang ada maka print normalized
      confusion matrix
9     else:
10         print('Confusion matrix, without normalization') #jika
      tidak memenuhi kondisi if maka pring else
11
12
13

```

```

14 print(cm) #print data cm
15
16 plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt sebagai fungsi untuk membuat plot
17 plt.title(title) #membuat title pada plot
18 #plt.colorbar()
19 tick_marks = np.arange(len(classes)) #membuat marks pada plot
20 plt.xticks(tick_marks, classes, rotation=90) #membuat ticks pada x
21 plt.yticks(tick_marks, classes) #membuat ticks pada y
22
23 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
24 thresh = cm.max() / 2. #variabel thresh menambil data max pada cm kemudian dibagi 2
25
26 plt.tight_layout() #mengatur layout pada plot
27 plt.ylabel('True label') #memberi nama label pada sumbu y
28 plt.xlabel('Predicted label') #memberi nama label pada sumbu x
29
30
31 # In [22]:
32
33 birds = pd.read_csv("data/CUB_200_2011/classes.txt",
34                      sep='\s+', header=None, usecols=[1], names
35                      =[['birdname']]) #membaca csv dengan ketentuan nama birdname
36 birds = birds[['birdname']] #nama birds dengan ketentuan
37         birdname
38 birds #menampilkan data birds
39
40
41 # In [23]:
42
43 import numpy as np #import library numpy sebagai np
44 np.set_printoptions(precision=2) #np sebagai variabel yang
        membuat set precision=2
45 plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure
        dengan ketentuan sizw 60,60 dan dpi 300
46 plot_confusion_matrix(cm, classes=birds, normalize=True) #data
        cm dan clas birds dibuat sebagai plot

```

6. Jalankan program klasifikasi SVM dan Decission Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Menunjukkan klarifikasi dengan decission tree menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 from sklearn import tree #import library tree
2 clftree = tree.DecisionTreeClassifier() #clftree sebagai
        variabel untuk decision tree
3 clftree.fit(df_train_att, df_train_label) #sebagai data
        training

```

```
4 clftree.score(df_test_att, df_test_label) #sebagai data
      testing
```

- Menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```
1 from sklearn import svm #import library svm
2 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
      fungsi SVC
3 clfsvm.fit(df_train_att, df_train_label) #sebagai data
      training
4 clfsvm.score(df_test_att, df_test_label) #sebagai data testing
```

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

- Menunjukkan hasil cross validation pada Random Forest.

```
1 from sklearn.model_selection import cross_val_score #import
      cross_val_score
2 scores = cross_val_score(clf, df_train_att, df_train_label, cv
      =5) #variabel scores sebagai variabel prediksi dari data
      training
3 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
      std() * 2)) #print data scores dengan ketentuan akurasi
```

- Menunjukkan hasil cross validation pada Desicion Tree.

```
1 scorestree = cross_val_score(clftree, df_train_att,
      df.train_label, cv=5) #sebagai prediksi menggunakan scores
      dan metode tree
2 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
      scorestree.std() * 2)) #menampilkan dengan ketentuan yang
      ada
```

- Menunjukkan hasil cross validation pada SVM.

```
1 scoressvm = cross_val_score(clfsvm, df_train_att,
      df.train_label, cv=5) #sebagai data training
2 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
      scoressvm.std() * 2)) #sebagai data testing dan output
      akurasi
```

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

- Menunjukkan informasi-informasi tree yang dibuat.

```

1 max_features_opts = range(5, 50, 5) #max_features_opts sebagai
   variabel untuk membuat range 5,50,5
2 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
   sebagai variabel untuk membuat range 10,200,20
3 rf_params = np.empty((len(max_features_opts)*len(
   n_estimators_opts),4), float) #rf_params sebagai variabel
   untuk menjumlahkan yang sudah di tentukan sebelumnya
4 i = 0
5 for max_features in max_features_opts: #pengulangan
6   for n_estimators in n_estimators_opts: #pengulangan
7     clf = RandomForestClassifier(max_features=max_features
      , n_estimators=n_estimators) #menampilkan variabel csf
8     scores = cross_val_score(clf, df_train_att,
9       df_train_label, cv=5) #scores sebagai variabel training
10    rf_params[i,0] = max_features #index 0
11    rf_params[i,1] = n_estimators #index 1
12    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:
   %.02f (+/- %.02f)" % (max_features, n_estimators,
   scores.mean(), scores.std() * 2))

```

- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```

1 import matplotlib.pyplot as plt #import library matplotlib
   sebagai plt
2 from mpl_toolkits.mplot3d import Axes3D #import axes3D untuk
   menampilkan plot 3 dimensi
3 from matplotlib import cm #memanggil data cm yang sudah
   tersedia
4 fig = plt.figure() #hasil plot sebagai figure
5 fig.clf() #figure di ambil dari clf
6 ax = fig.gca(projection='3d') #ax sebagai projection 3d
7 x = rf_params[:,0] #x sebagai index 0
8 y = rf_params[:,1] #y sebagai index 1
9 z = rf_params[:,2] #z sebagai index 2
10 ax.scatter(x, y, z) #membuat plot scatter x y z
11 ax.set_zlim(0.2, 0.5) #set zlim dengan ketentuan yang ada
12 ax.set_xlabel('Max features') #memberi nama label x
13 ax.set_ylabel('Num estimators') #memberi nama label y
14 ax.set_zlabel('Avg accuracy') #memberi nama label z
15 plt.show() #print hasil plot yang sudah dibuat.

```

### 3.5.3 Penanganan Error

#### 1. ScreenShoot Error

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

**Gambar 3.92** FileNotFoundError

#### 2. Tuliskan Kode Error dan Jenis Error

- FileNotFoundException

### 3. Cara Penangan Error

- FileNotFoundException

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.

#### 3.5.4 Bukti Tidak Plagiat



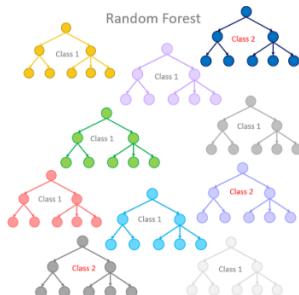
Gambar 3.93 FileNotFoundException

### 3.6 1174008 - Arjun Yuda Firwanda

#### 3.6.1 Soal Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest merupakan suatu metode penyelesaian permasalahan. Metode Random Forest sebagai sebutan metode Decision Tree atau metode yang menyerupai pohon pengambilan keputusan sebuah diagram air yang memiliki sebuah root node yang digunakan untuk mengumpulkan suatu data.



Gambar 3.94 Random Forest.

2. Jelaskan 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. Jelaskan apa itu cross validation.

Cross Validation suatu metode yang menerapkan statistik yang biasanya dapat digunakan untuk menilai kinerja model serta algoritma, Pada data tersebut dapat dipisahkan menjadi dua bagian subset, seperti data proses pembelajaran dan data validasi atau evaluasi. Selain itu juga Cross Validation ini dapat digunakan dikarenakan mampu mengurangi waktu komputasi dengan tetap serta dapat menjaga keakuratan estimasi.

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
				Data Pengujian					
				Data Pelatihan					

Gambar 3.95 Random Forest.

4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh 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 decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision 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.

5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

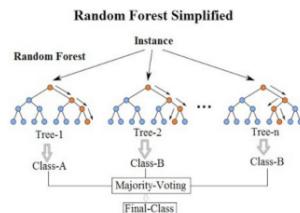
Confusion Matrix atau error matrix yang dapat memberikan informasi perbandingan hasil klasifikasi sebenarnya. Confusion Matrix berbentuk tabel matriks yang menggambarkan kinerja model klasifikasi pada serangkaian data uji yang nilai sebenarnya telah diketahui.

		Actual Values	
		1 (Positive)	0 (Negative)
Predicted Values	1 (Positive)	TP (True Positive)	FP (False Positive) <small>Type I Error</small>
	0 (Negative)	FN (False Negative) <small>Type II Error</small>	TN (True Negative)

Gambar 3.96 Random Forest.

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri

Voting pada random forest ialah sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Penentuan klasifikasi menggunakan metode random forest diambil berdasarkan hasil voting dari decision tree yang terbentuk. Setelah decision tree atau pohon terbentuk, maka akan dilakukan voting pada setiap kelas dari data sampel. Kemudian, selanjutnya mengkombinasikan vote dari setiap kelas kemudian diambil vote yang paling banyak. Dengan menggunakan metode penyelesaian random forest pada klasifikasi data maka, akan menghasilkan vote yang paling baik.



Gambar 3.97 Random Forest.

### 3.6.2 Praktek Program

- Buat aplikasi sederhana menggunakan pandas dan jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas).

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 10:34:35 2020
4
5 @author: Arjun

```

Kode di atas digunakan untuk mengimpor atau mengirim library pandas sebagai Arjun, Hasilnya adalah sebagai berikut :

```

In [5]: import pandas as Arjun #melakukan import library pandas sebagai arjun
.....
.... laptop = {"Nama Laptop": ['Asus', 'ROG', 'Lenovo', 'Samsung']} membuat varibel
.... yang bernama laptop , dan mengisi dataframe
.... x = Arjun.DataFrame(laptop) membuat x membuat DataFrame dari library pandas
.... dan mengisi data
.... print ('Arjun Punya Laptop ' + x) print hasil dari x
0 Arjun Punya Laptop Asus
1 Arjun Punya Laptop ROG
2 Arjun Punya Laptop Lenovo
3 Arjun Punya Laptop Samsung

```

**Gambar 3.98** Hasil Soal 1.

- buat aplikasi sederhana menggunakan numpy dan jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```

1
2 laptop = {"Nama Laptop": ['Asus', 'ROG', 'Lenovo', 'Samsung']} # membuat varibel yang bernama laptop , dan mengisi dataframe nama2 laptop
3 x = Arjun.DataFrame(laptop) #variabel x membuat DataFrame dari library pandas dan akan memanggil variabel laptop.
4 print (' Arjun Punya Laptop ' + x) #print hasil dari x
5
6 # In [44]: Soal2

```

Fungsi eye disini berguna untuk memanggil matrix identitas dengan jumlah colom dan baris sesuai yang ditentukan. Disini saya telah menentukan 10 colom dan baris maka dari itu hasilnya akan memunculkan matrix identitas 10x10, Hasilnya adalah sebagai berikut :

```

In [5]: import numpy as Arjun #melakukan import numpy sebagai arjun
.....
.... matrix_x = Arjun.eye(10) membuat matrix dengan numpy dengan menggunakan fungsi eye
.....
.... matrix_x = Arjun.crosstab(matrix_x, matrix_x) membuat matrix_x yang telah dilakukan
.....
.... print (matrix_x) print matrix_x yang telah dilakukan dengan 10x10
[[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 3.99** Hasil Soal 2.

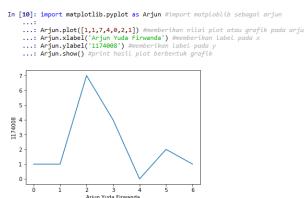
- buat aplikasi sederhana menggunakan matplotlib dan jelaskan arti dari setiap baris kode(harus beda dengan teman sekelas)

```

1 matrix_x # deklarasikan matrix_x yang telah dibuat
2
3 print (matrix_x) # print matrix_x yang telah dibuat dengan 10x10
4
5
6 # In [44]: Soal3

```

Jalankan source code diatas dengan spyder atau anaconda sehingga hasilnya akan seperti berikut :



**Gambar 3.100** Hasil Soal 3.

4. jalankan program klasifikasi Random Forest pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- 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.

```

1 # In [1]: RANDOM FOREST
2
3 import pandas as pd #import library pandas sebagai as
4
5 imgatt = pd.read_csv("data/CUB_200_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']) #library imgatt sebagai membaca file csv dari
dataset , dengan ketentuan yang ada.

```

Hasilnya adalah seperti ini :

```

In [1]: Import pandas as pd
...:
...: imgatt = pd.read_csv("data/CUB_200_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']) #library imgatt sebagai membaca file csv dari
dataset , dengan ketentuan yang ada.

```

**Gambar 3.101** Hasil Soal 4 - 1

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

```
1 # In [2]:  
2  
3 imgatt.head() #menampilkan data yang di baca tadi tetapi hanya  
    data paling atas
```

Hasilnya adalah seperti ini :

	In [2]:	imgatt.head()
	Out[2]:	
0	imgid	attid
1	1	1
2	1	2
3	1	3
4	1	4
		0
		0
		0
		1

**Gambar 3.102** Hasil Soal 4 - 2

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

```
1 # In [3]:  
2  
3 imgatt.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

	In [3]:	imgatt.shape
	Out[3]:	(3677856, 3)

**Gambar 3.103** Hasil Soal 4 - 3

- Variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya.

```
1 # In [4]:  
2  
3 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present') #membuat variabel baru dari fungsi imgatt ,  
    dengan mengganti index dan kolom (kebalikan)
```

Hasilnya adalah seperti ini :

In [4]:	imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
---------	--

**Gambar 3.104** Hasil Soal 4 - 4

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

```

1 # In [5]:
2
3 imgatt2.head() #menampilkan data yang di baca tadi tetapi
    hanya data paling atas

```

Hasilnya adalah seperti ini :

	In [5]:	Out[5]:
1	<code>imgatt2.head()</code>	<code>imgatt2</code>
2		sizeid 1 2 3 4 5 6 7 ... 386 387 388 389 310 311 312
3		imgid 0 0 0 0 0 0 0 ... 0 0 1 0 0 0 0
4		0 0 0 0 0 0 0 ... 0 0 0 1 0 0 0
5		0 0 0 0 0 0 1 0 0 ... 0 0 0 0 1 0 0
		[5 rows x 312 columns]

**Gambar 3.105** Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```

1 # In [6]:
2
3 imgatt2.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

	In [6]:	Out[6]:
1	<code>imgatt2.shape</code>	<code>(11788, 312)</code>

**Gambar 3.106** Hasil Soal 4 - 6

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

```

1 # In [7]:
2
3 imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
4                         sep=' ', header=None, names=['imgid',
5                         'label']) #baca data csv dengan ketentuan yang ada
6
6 imglabels = imglabels.set_index('imgid') #variabel imglabels
    sebagai set index (imgid)

```

Hasilnya adalah seperti ini :

	In [8]:	Out[8]:
1	<code>imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",         sep=' ', header=None, names=['imgid', 'label'])</code>	<code># a description from dataset README:</code>
2		<code>## the ground truth class labels (bird species labels) for each image are contained</code>
3		<code>## in the file image_class_labels.txt. The file contains two columns: image_id and</code>
4		<code>## class_id. image_id is image_id and class_id is class_label.</code>
5		<code>## image_id and class_id correspond to the IDs in images.txt and classes.txt,</code>
6		<code>## respectively.</code>
7	<code>imglabels.head()</code>	<code>Out[8]:</code>
8		<code>imgid</code>
9		<code>1</code>
10		<code>2</code>
11		<code>3</code>
12		<code>4</code>
13		<code>5</code>

**Gambar 3.107** Hasil Soal 4 - 7

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

```

1 # In [8]:
2
3 imglabels.head() #menampilkan data yang di baca tadi tetapi
      hanya data paling atas

```

Hasilnya adalah seperti ini :

In [9]:	imglabels.head()
Out[9]:	
	label
	imgid
1	1
2	1
3	1
4	1
5	1

**Gambar 3.108** Hasil Soal 4 - 8

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

```

1 # In [9]:
2
3 imglabels.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

In [10]:	imglabels.shape
Out[10]:	(11788, 1)

**Gambar 3.109** Hasil Soal 4 - 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 # In [10]:
2
3 df = imgatt2.join(imglabels) #variabel df sebagai fungsi join
      dari data imgatt2 ke variabel imglabels
4 df = df.sample(frac=1) #variabel df sebagai sample dengan
      ketentuan frac=1

```

Hasilnya adalah seperti ini :

In [11]:	df = imgatt2.join(imglabels)
...:	df = df.sample(frac=1)

**Gambar 3.110** Hasil Soal 4 - 10

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

```

1 # In [11]:
2
3 df_att = df.iloc[:, :312] #membuat kolom dengan ketentuan 312
4 df_label = df.iloc[:, 312:] #membuat kolom dengan ketentuan
    312

```

Hasilnya adalah seperti ini :

```

In [12]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]

```

**Gambar 3.111** Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```

1 # In [12]:
2
3 df_att.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas

```

Hasilnya adalah seperti ini :

```

In [13]: df_att.head()
Out[13]:
   imgid  2  3  4  5  6  7  ...  366  397  388  369  319  311  312
0  2449  0  0  0  0  0  1  ...  0  0  0  0  0  0  0  1
1  988   0  0  0  0  0  0  0  ...  0  0  0  0  0  0  0  0
2  7472  0  0  0  0  0  0  0  ...  0  0  0  0  0  0  0  1
3  7084  0  0  0  0  0  0  1  ...  0  0  0  0  0  1  0  0
4  9834  0  0  0  0  0  0  0  ...  0  0  1  0  0  0  0  1
[5 rows x 312 columns]

```

**Gambar 3.112** Hasil Soal 4 - 12

- Mengecek isi data teratas dari df label.

```

1 # In [13]:
2
3 df_label.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas

```

Hasilnya adalah seperti ini :

```

In [14]: df_label.head()
Out[14]:
label
imgid
2449      43
988       18
7472      128
7084      121
9834      168

```

**Gambar 3.113** Hasil Soal 4 - 13

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

```

1 # In [14]:
2
3 df_train_att = df_att[:8000] #akan membagi 8000 row pertama
   menjadi data training dan sisanya adalah data testing
4 df_train_label = df_label[:8000] #akan membagi 8000 row
   pertama menjadi data training dan sisanya adalah data
   testing
5 df_test_att = df_att[8000:] #kebalikan dari akan membagi 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
6 df_test_label = df_label[8000:] #kebalikan dari akan membagi
   8000 row pertama menjadi data training dan sisanya adalah
   data testing
7
8 df_train_label = df_train_label['label'] #menampilkan data
   training
9 df_test_label = df_test_label['label'] #menampilkan data
   testing

```

Hasilnya adalah seperti ini :

```

In [15]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']

```

**Gambar 3.114** Hasil Soal 4 - 14

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

```

1 # In [15]:
2
3 from sklearn.ensemble import RandomForestClassifier #import
   randomforestclassifier
4 clf = RandomForestClassifier(max_features=50, random_state=0,
   n_estimators=100) #clf sebagai variabel untuk klafisikasi
   random forest

```

Hasilnya adalah seperti ini :

```

In [16]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)

```

**Gambar 3.115** Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 # In [16]:
2
3 clf.fit(df_train_att, df_train_label) #variabel clf untuk fit
   yaitu training

```

Hasilnya adalah seperti ini :

```
In [12]: clf.fit(df_train_att, df_train_label)
Out[12]: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                           criterion='gini', max_depth=None, max_features=50,
                           max_leaf_nodes=None, max_memory=312500000.0,
                           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=-1, oob_score=False, random_state=42,
                           verbose=0, warm_start=False)
```

**Gambar 3.116** Hasil Soal 4 - 16

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

```
1 # In [17]:
2
3 print(clf.predict(df_train_att.head())) #print clf yang di
    prediksi dari training tetapi hanya menampilkan data
    paling atas
```

Hasilnya adalah seperti ini :

```
In [18]: print(clf.predict(df_train_att.head()))
[ 43 18 128 121 168]
```

**Gambar 3.117** Hasil Soal 4 - 17

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

```
1 # In [18]:
2
3 clf.score(df_test_att, df_test_label) #print clf sebagai
    testing yang sudah di training tadi
```

Hasilnya adalah seperti ini :

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.4390179514255544
```

**Gambar 3.118** Hasil Soal 4 - 18

- Jalankan program confusion matrix pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Melakukan import confusion matrix pada library sklearn matriks.

```
1 # In [19]: Confusion Matrix
2
3 from sklearn.metrics import confusion_matrix #import Confusion
    Matrix
4 pred_labels = clf.predict(df_test_att) #sebagai data testing
5 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
    variabel data label
```

Hasilnya adalah seperti ini :

```
In [20]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_xty)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

**Gambar 3.119** Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```
1 # In [20]:
2
3 cm #menampilkan data label berbentuk array
```

Hasilnya adalah seperti ini :

```
In [21]: cm
Out[21]:
array([[ 8,  1,  1, ...,  0,  1,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 0,  1,  6, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  1,  0,  0],
       [ 0,  0,  0, ...,  0,  5,  0],
       [ 0,  0,  0, ...,  0,  0,  1]], dtype=int64)
```

**Gambar 3.120** Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```
1 # In [21]:
2
3 import matplotlib.pyplot as plt #import library matplotlib
4         sebagai plt
5 import itertools #import library itertools
6 def plot_confusion_matrix(cm, classes,
7                           normalize=False,
8                           title='Confusion matrix',
9                           cmap=plt.cm.Blues): #membuat fungsi
10    dengan ketentuan data yang ada pada cm
11
12    if normalize:
13        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
14    ]
15        print("Normalized confusion matrix") #jika normalisasi
16        sebagai ketentuan yang ada maka print normalized
17        confusion matrix
18    else:
19        print('Confusion matrix, without normalization') #jika
20        tidak memenuhi kondisi if maka pring else
21
22    print(cm) #print data cm
23
24    plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
25    sebagai fungsi untuk membuat plot
26    plt.title(title) #membuat title pada plot
27    #plt.colorbar()
28    tick_marks = np.arange(len(classes)) #membuat marks pada
29    plot
```

```
22 plt.xticks(tick_marks, classes, rotation=90) #membuat  
23 ticks pada x  
24  
25 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi  
26 thresh = cm.max() / 2. #variabel thresh menambil data max  
27 pada cm kemudian dibagi 2  
28  
29 plt.tight_layout() #mengatur layout pada plot  
30 plt.ylabel('True label') #memberi nama label pada sumbu y  
31 plt.xlabel('Predicted label') #memberi nama label pada  
32 sumbu x  
33  
34 # In [22]:  
35  
36 birds = pd.read_csv("data/CUB_200_2011/classes.txt",  
37 sep='\s+', header=None, usecols=[1], names  
38 =['birdname']) #membaca csv dengan ketentuan nama birdname  
39 birds = birds['birdname'] #nama birds dengan ketentuan  
40 birdname  
41 birds #menampilkan data birds  
42  
43 # In [23]:  
44  
45 import numpy as np #import library numpy sebagai np  
46 np.set_printoptions(precision=2) #np sebagai variabel yang  
47 membuat set precision=2  
48 plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure  
49 dengan ketentuan sizw 60,60 dan dpi 300  
50 plot_confusion_matrix(cm, classes=birds, normalize=True) #data  
51 cm dan clas birds dibuat sebagai plot  
52 plt.show() #menampilkan hasil plot yang berbentuk grafik
```

Hasilnya adalah seperti dibawah ini :

```
[2]: import numpy as np
import tensorflow as tf
def plot_confusion_matrix(cm, classes,
                         normalize=False,
                         title='Confusion matrix',
                         cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
    else:
        print('without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.

    for i, j in np.ndindex(cm.shape):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 verticalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

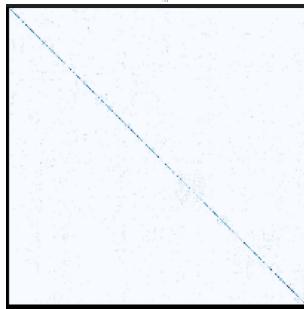
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

[3]: X_train, y_train, X_test, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

plot_confusion_matrix(tf.argmax(y_train, axis=1), np.arange(10))
plot_confusion_matrix(tf.argmax(y_test, axis=1), np.arange(10))

[4]: [In 25]: b1ds = pd.read_csv("data/b1ds.csv")
b1ds.info()

Out[25]:<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 10 columns):
 #   Column  Non-Null Count  Dtype  
 --- 
 0   id      100 non-null   int64  
 1   house_id 100 non-null   object  
 2   street   100 non-null   object  
 3   building 100 non-null   object  
 4   room_type 100 non-null   object  
 5   neighborhood 100 non-null   object  
 6   city     100 non-null   object  
 7   country  100 non-null   object  
 8   price    100 non-null   float64
 9   date     100 non-null   datetime64[ns]
dtypes: datetime64[ns](1), float64(1), int64(1), object(7)
memory usage: 8.0 KB
```



**Gambar 3.121** Menampilkan hasil plot

6. Jalankan program klasifikasi SVM dan Decission Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Menunjukkan klarifikasi dengan decision tree menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```
1 # In [24]: Mencoba dengan metode Decission Tree dan SVM
2
3 from sklearn import tree #import library tree
4 clftree = tree.DecisionTreeClassifier() #clftree sebagai
   variabel untuk decision tree
5 clftree.fit(df_train_att, df_train_label) #sebagai data
   training
6 clftree.score(df_test_att, df_test_label) #sebagai data
   testing
```

Hasilnya adalah seperti ini :

```
In [27]: from sklearn import tree
      ... clftree = tree.DecisionTreeClassifier()
      ... clftree.fit(df_train_att, df_train_label)
      ... clftree.score(df_test_att, df_test_label)
Out[27]: 0.26135163674762407
```

**Gambar 3.122** Hasil Soal 6 - 1

- Menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 # In [25]:
2
3 from sklearn import svm #import library svm
4 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
      fungsi SVC
5 clfsvm.fit(df_train_att, df_train_label) #sebagai data
      training
6 clfsvm.score(df_test_att, df_test_label) #sebagai data testing

```

Hasilnya adalah seperti ini :

```

In [28]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
Out[28]: 0.47729672650475186

```

**Gambar 3.123** Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

```

1 # In [26]: Pengecekan Cross Validation
2
3 from sklearn.model_selection import cross_val_score #import
      cross_val_score
4 scores = cross_val_score(clf, df_train_att, df_train_label, cv
      =5) #variabel scores sebagai variabel prediksi dari data
      training
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
      std() * 2)) #print data scores dengan ketentuan akurasi

```

Hasilnya adalah seperti ini :

```

In [29]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.45 (+/- 0.03)

```

**Gambar 3.124** Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

```

1 # In [27]:
2
3 scorestree = cross_val_score(clftree, df_train_att,
      df_train_label, cv=5) #sebagai prediksi menggunakan scores
      dan metode tree
4 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
      scorestree.std() * 2)) #menampilkan dengan ketentuan yang
      ada

```

Hasilnya adalah seperti ini :

```
[In [28]: scoretree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoretree.mean(), scoretree.std()))
[2]: Accuracy: 0.26 (+/- 0.01)
```

**Gambar 3.125** Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

```
1 # In [28]:
2
3 scoressvm = cross_val_score(clfsvm, df_train_att,
4     df_train_label, cv=5) #sebagai data training
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
6     scoressvm.std() * 2)) #sebagai data testing dan output
7 akurasi
```

Hasilnya adalah seperti ini :

```
[In [28]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
[2]: Accuracy: 0.47 (+/- 0.01)
```

**Gambar 3.126** Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

```
1 # In [29]: Pengamatan komponen informasi
2
3 max_features_opts = range(5, 50, 5) #max_features_opts sebagai
4     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 = np.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         clf = RandomForestClassifier(max_features=max_features
14             , n_estimators=n_estimators) #menampilkan variabel csf
15         scores = cross_val_score(clf, df_train_att,
16             df_train_label, cv=5) #scores sebagai variabel training
17         rf_params[i,0] = max_features #index 0
18         rf_params[i,1] = n_estimators #index 1
19         rf_params[i,2] = scores.mean() #index 2
20         rf_params[i,3] = scores.std() * 2 #index 3
21         i += 1 #dengan ketentuan i += 1
22         print("Max features: %d, num estimators: %d, accuracy:
23             %0.2f (+/- %0.2f)" % (max_features, n_estimators,
24             scores.mean(), scores.std() * 2))
25         #print hasil pengulangan yang sudah ditentukan
```

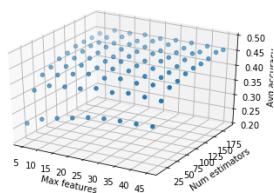
Hasilnya adalah seperti ini :

**Gambar 3.127** Hasil Soal 8 - 1

- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```
1 # In [30]:  
2  
3 import matplotlib.pyplot as plt #import library matplotlib  
4         sebagai plt  
4 from mpl_toolkits.mplot3d import Axes3D #import axes3D untuk  
5         menampilkan plot 3 dimensi  
5 from matplotlib import cm #memanggil data cm yang sudah  
6         tersedia  
6 fig = plt.figure() #hasil plot sebagai figure  
7 fig.clf() #figure di ambil dari clf  
8 ax = fig.gca(projection='3d') #ax sebagai projection 3d  
9 x = rf_params[:,0] #x sebagai index 0  
10 y = rf_params[:,1] #y sebagai index 1  
11 z = rf_params[:,2] #z sebagai index 2  
12 ax.scatter(x, y, z) #membuat plot scatter x y z  
13 ax.set_zlim(0.2, 0.5) #set zlim dengan ketentuan yang ada  
14 ax.set_xlabel('Max features') #memberi nama label x  
15 ax.set_ylabel('Num estimators') #memberi nama label y  
16 ax.set_zlabel('Avg accuracy') #memberi nama label z  
17 plt.show() #print hasil plot yang sudah dibuat.
```

Hasilnya adalah seperti ini:



**Gambar 3.128** Hasil Soal 8 - 2

### 3.6.3 Penanganan Error

#### 1. ScreenShoot Error



```
F:\data\error> [Error] F:\data\b3\b3-data\b3_300_2011\atributes\image_attributes_labels.txt: file does not exist.
b3-data\b3\b3_300_2011\atributes\image_attributes_labels.txt"
```

**Gambar 3.129** FileNotFoundError

#### 2. Tuliskan Kode Error dan Jenis Error

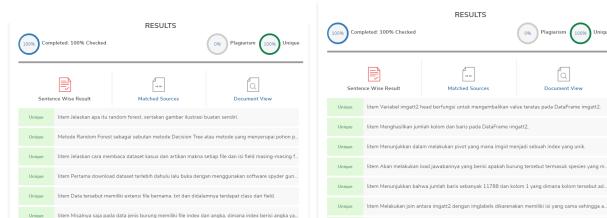
- FileNotFoundError

#### 3. Cara Penangan Error

- FileNotFoundError

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.

### 3.6.4 Bukti Tidak Plagiat



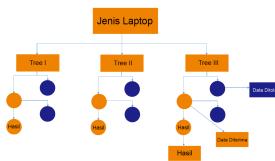
**Gambar 3.130** Bukti Tidak Melakukan Plagiat Chapter 3

## 3.7 1174026 - Felix Lase

### 3.7.1 Soal Teori

#### 1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest merupakan 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. Untuk gambar Random Forest dapat dilihat dibawah ini :



**Gambar 3.131** Random Forest.

2. Jelaskan 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. Jelaskan apa itu cross validation.

Cross Validation merupakan 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 mengujinya dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh 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 decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision 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.

- Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

1 fig = plt.figure() #hasil plot sebagai figure
2 fig.clf() #figure di ambil dari clf
3 ax = fig.gca(projection='3d') #ax sebagai projection 3d
4 x = rf_params[:,0] #x sebagai index 0
5 y = rf_params[:,1] #y sebagai index 1

```

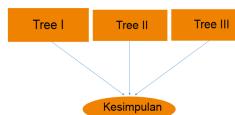
Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data positif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data positif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat dibawah ini :



**Gambar 3.132** Confusion Matriks.

- Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri

Voting pada random forest ialah sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk lebih jelasnya saya akan memberikan sebuah contoh bagaimana voting bekerja :



**Gambar 3.133** Decision Tree.

Dimana ditunjukkan pada gambar diatas terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan diadakan voting untuk menentukan sebuah laptop. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan laptop asus, yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah laptop asus. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menyatakan laptop asus sedangkan pada tree 3 menyatakan laptop HP, maka kesimpulan yang diambil adalah laptop asus dikarenakan hasil voting terbanyak adalah laptop asus.

### 3.7.2 Praktek Program

1. Buat aplikasi sederhana menggunakan pandas dan jelaskan arti setiap baris kode yang dibuat(harus beda dengan teman sekelas).

```

1 # In[44]: Soal1
2
3 import pandas as felix #melakukan import pada library pandas
4         sebagai felix
5 laptop = {"Nama Laptop" : ['Asus', 'HP', 'Lenovo', 'Samsung']} # membuat varibel yang bernama laptop, dan mengisi dataframe
6         nama2 laptop
    
```

Kode di atas digunakan untuk mengimpor atau mengirim library pandas sebagai fahmi, Hasilnya adalah sebagai berikut :

	Nama Laptop
0	Fahmi Punya Laptop Asus
1	Fahmi Punya Laptop HP
2	Fahmi Punya Laptop Lenovo
3	Fahmi Punya Laptop Samsung

**Gambar 3.134** Hasil Soal 1.

2. buat aplikasi sederhana menggunakan numpy dan jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)

```

1 import numpy as felix #melakukan import numpy sebagai felix
2
3 matrix_x = felix.eye(10) #membuat matrix dengan numpy dengan
4         menggunakan fungsi eye
5 matrix_x #deklrasikan matrix_x yang telah dibuat
6
6 print (matrix_x) #print matrix_x yang telah dibuat dengan 10x10
    
```

Fungsi eye disini berguna untuk memanggil matrix identitas dengan jumlah kolom dan baris sesuai yang ditentukan. Disini saya telah menentukan 10 kolom

dan baris maka dari itu hasilnya akan memunculkan matrix identitas 10x10, Hasilnya adalah sebagai berikut :

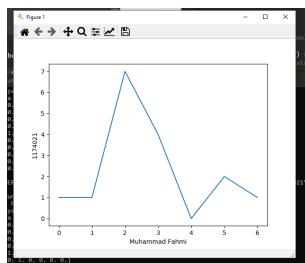
```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]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

**Gambar 3.135** Hasil Soal 2.

3. buat aplikasi sederhana menggunakan matplotlib dan jelaskan arti dari setiap baris kode(harus beda dengan teman sekelas)

```
1 import matplotlib.pyplot as felix #import matplotlib sebagai felix
2
3 felix.plot([1,1,7,4,0,2,1]) #memberikan nilai plot atau grafik pada felix
4 felix.xlabel('Muhammad felix') #memberikan label pada x
5 felix.ylabel('1174026') #memberikan label pada y
6 felix.show() #print hasil plot berbentuk grafik
```

Jalankan source code diatas dengan spyder atau anaconda sehingga hasilnya akan seperti berikut :



**Gambar 3.136** Hasil Soal 3.

4. jalankan program klasifikasi Random Forest pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- 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.

```

1 # In [1]: RANDOM FOREST
2
3 import pandas as pd #import library pandas sebagai as
4
5 imgatt = pd.read_csv("data/CUB_200_2011/attributes/
6     image_attribute_labels.txt",
7                         sep='|', header=None, error_bad_lines=
8     False, warn_bad_lines=False,
9                         usecols=[0,1,2], names=['imgid', 'attid',
10                         'present']) #library imgatt sebagai membaca file csv dari
11 dataset, dengan ketentuan yang ada.

```

Hasilnya adalah seperti ini :

**Gambar 3.137** Hasil Soal 4 - 1

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

```

1 # In [2]:
2
3 imgatt.head() #menampilkan data yang di baca tadi tetapi hanya
               data paling atas

```

Hasilnya adalah seperti ini :

imgid	attid	present
0	1	0
1	2	0
2	3	0
3	4	0
4	5	1

**Gambar 3.138** Hasil Soal 4 - 2

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

```

1 # In [3]:
2
3 imgatt.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

	In [3]: imgatt.shape
	Out[3]: (3677856, 3)

**Gambar 3.139** Hasil Soal 4 - 3

- Variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya.

```
1 # In [4]:
2
3 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values=
   'present') #membuat variabel baru dari fungsi imgatt,
   dengan mengganti index dan kolom (kebalikan)
```

Hasilnya adalah seperti ini :

In [4]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
--

**Gambar 3.140** Hasil Soal 4 - 4

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

```
1 # In [5]:
2
3 imgatt2.head() #menampilkan data yang di baca tadi tetapi
   hanya data paling atas
```

Hasilnya adalah seperti ini :

In [5]: imgatt2.head()
Out[5]:
imgatt2
imgid
1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
0 0 0 0 0 0 0 ... 0 0 1 0 0 0 0
0 0 0 0 0 0 0 ... 0 0 0 1 0 0 0
0 0 0 0 0 0 0 ... 0 0 0 0 1 0 0
0 0 0 0 0 0 0 ... 0 0 0 0 0 1 0
0 0 0 0 0 0 0 ... 0 0 0 0 0 0 1
[5 rows x 312 columns]

**Gambar 3.141** Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 # In [6]:
2
3 imgatt2.shape #menampilkan jumlah data , kolom
```

Hasilnya adalah seperti ini :

	In [6]: imgatt2.shape
	Out[6]: (11788, 312)

**Gambar 3.142** Hasil Soal 4 - 6

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

```

1 # In [7]:
2
3 imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
4                         sep=' ', header=None, names=['imgid',
5                         'label']) #baca data csv dengan ketentuan yang ada
6
7 imglabels = imglabels.set_index('imgid') #variabel imglabels
8         sebagai set index (imgid)

```

Hasilnya adalah seperti ini :

```

In [8]: imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
... sep=' ', header=None, names=['imgid', 'label'])
...
Out[8]: imglabels = imglabels.set_index('imgid')
...
In [9]: imglabels.head()
...
Out[9]: imglabels.head()
          label
imgid
1           1
2           1
3           1
4           1
5           1

```

**Gambar 3.143** Hasil Soal 4 - 7

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

```

1 # In [8]:
2
3 imglabels.head() #menampilkan data yang di baca tadi tetapi
4         hanya data paling atas

```

Hasilnya adalah seperti ini :

```

In [9]: imglabels.head()
Out[9]:
          label
imgid
1           1
2           1
3           1
4           1
5           1

```

**Gambar 3.144** Hasil Soal 4 - 8

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

```

1 # In [9]:
2
3 imglabels.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

In [10]:	imglabels.shape
Out[10]:	(11788, 1)

**Gambar 3.145** Hasil Soal 4 - 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 # In [10]:
2
3 df = imgatt2.join(imglabels) #variabel df sebagai fungsi join
    dari data imgatt2 ke variabel imglabels
4 df = df.sample(frac=1) #variabel df sebagai sample dengan
    ketentuan frac=1

```

Hasilnya adalah seperti ini :

In [11]:	df = imgatt2.join(imglabels)
...:	df = df.sample(frac=1)

**Gambar 3.146** Hasil Soal 4 - 10

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

```

1 # In [11]:
2
3 df_att = df.iloc[:, :312] #membuat kolom dengan ketentuan 312
4 df_label = df.iloc[:, 312:] #membuat kolom dengan ketentuan
    312

```

Hasilnya adalah seperti ini :

In [12]:	df_att = df.iloc[:, :312]
...:	df_label = df.iloc[:, 312:]

**Gambar 3.147** Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```

1 # In [12]:
2
3 df_att.head() #menampilkan data yang di baca tadi tetapi hanya
    data paling atas

```

Hasilnya adalah seperti ini :

In [13]:	df_att.head()																																																																																																
Out[13]:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>imgId</th> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> <th>6</th> <th>7</th> <th>...</th> <th>306</th> <th>307</th> <th>308</th> <th>309</th> <th>310</th> <th>311</th> <th>312</th> </tr> </thead> <tbody> <tr><td>2449</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>930</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>7472</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>2064</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>...</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>9834</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>...</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> </tbody> </table>	imgId	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312	2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1	930	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	2064	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	9834	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1
imgId	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312																																																																																		
2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1																																																																																		
930	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																		
7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1																																																																																		
2064	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																		
9834	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1																																																																																		
	[5 rows x 312 columns]																																																																																																

**Gambar 3.148** Hasil Soal 4 - 12

- Mengecek isi data teratas dari df\_label.

```
1 # In [13]:
2
3 df_label.head() #menampilkan data yang di baca tadi tetapi
      hanya data paling atas
```

Hasilnya adalah seperti ini :

	In [14]: df_label.head()
Out[14]:	label
imgid	
2449	43
988	18
7472	128
7084	121
9834	168

**Gambar 3.149** Hasil Soal 4 - 13

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

```
1 # In [14]:
2
3 df_train_att = df_att[:8000] #akan membagi 8000 row pertama
      menjadi data training dan sisanya adalah data testing
4 df_train_label = df_label[:8000] #akan membagi 8000 row
      pertama menjadi data training dan sisanya adalah data
      testing
5 df_test_att = df_att[8000:] #kebalikan dari akan membagi 8000
      row pertama menjadi data training dan sisanya adalah data
      testing
6 df_test_label = df_label[8000:] #kebalikan dari akan membagi
      8000 row pertama menjadi data training dan sisanya adalah
      data testing
7
8 df_train_label = df_train_label['label'] #menampilkan data
      training
9 df_test_label = df_test_label['label'] #menampilkan data
      testing
```

Hasilnya adalah seperti ini :

	In [15]: df_train_att = df_att[:8000] ...: df_train_label = df_label[:8000] ...: df_test_att = df_att[8000:] ...: df_test_label = df_label[8000:] ...: ...: df_train_label = df_train_label['label'] ...: df_test_label = df_test_label['label']
--	--

**Gambar 3.150** Hasil Soal 4 - 14

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

```

1 # In [15]:
2
3 from sklearn.ensemble import RandomForestClassifier #import
4         randomforestclassifier
5 clf = RandomForestClassifier(max_features=50, random_state=0,
6     n_estimators=100) #clf sebagai variabel untuk klafifikasi
7         random forest

```

Hasilnya adalah seperti ini :

**Gambar 3.151** Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 # In [16]:
2
3 clf.fit(df_train_att, df_train_label) #variabel clf untuk fit
4         yaitu training

```

Hasilnya adalah seperti ini :

**Gambar 3.152** Hasil Soal 4 - 16

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

```

1 # In [17]:
2
3 print(clf.predict(df_train_att.head())) #print clf yang di
4         prediksi dari training tetapi hanya menampilkan data
5         paling atas

```

Hasilnya adalah seperti ini :

**Gambar 3.153** Hasil Soal 4 - 17

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

```

1 # In [18]:
2
3 clf.score(df_test_att, df_test_label) #print clf sebagai
4         testing yang sudah di training tadi

```

Hasilnya adalah seperti ini :

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.4390179514255544
```

**Gambar 3.154** Hasil Soal 4 - 18

- Jalankan program confusion matrix pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Melakukan import confusion matrix pada library sklearn matriks.

```
1 # In [19]: Confusion Matrix
2
3 from sklearn.metrics import confusion_matrix #import Confusion
4 Matrix
5 pred_labels = clf.predict(df_test_att) #sebagai data testing
6 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
7 variabel data label
```

Hasilnya adalah seperti ini :

```
In [20]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

**Gambar 3.155** Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```
1 # In [20]:
2
3 cm #menampilkan data label berbentuk array
```

Hasilnya adalah seperti ini :

```
In [21]: cm
Out[21]:
array([[ 8,  1,  1, ...,  0,  1,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 0,  1,  6, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  1,  0,  0],
       [ 0,  0,  0, ...,  0,  3,  0],
       [ 0,  0,  0, ...,  0, 16]], dtype=int64)
```

**Gambar 3.156** Hasil Soal 5 - 2

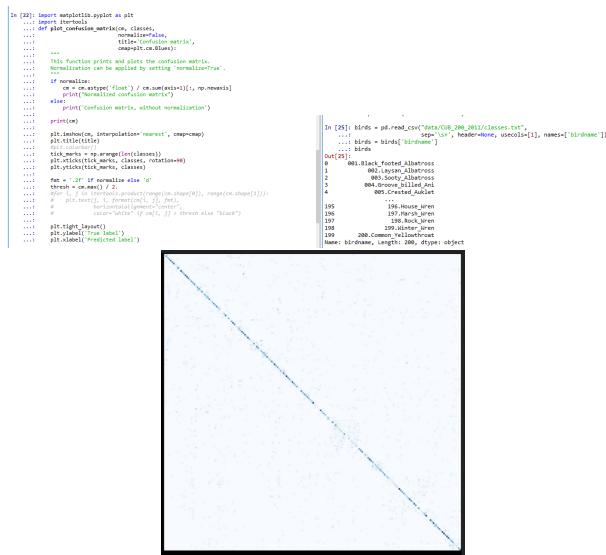
- Membuat dan menampilkan hasil plot.

```
1 # In [21]:
2
3 import matplotlib.pyplot as plt #import library matplotlib
4 sebagai plt
5 import itertools #import library itertools
6 def plot_confusion_matrix(cm, classes,
```

```
6             normalize=False ,
7             title='Confusion matrix ' ,
8             cmap=plt.cm.Blues): #membuat fungsi
9     dengan ketentuan data yang ada pada cm
10
11     if normalize:
12         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13
14         print("Normalized confusion matrix") #jika normalisasi
15         sebagai ketentuan yang ada maka print normalized
16         confusion matrix
17     else:
18         print('Confusion matrix , without normalization') #jika
19         tidak memenuhi kondisi if maka pring else
20
21     print(cm) #print data cm
22
23
24
25     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
26     sebagai fungsi untuk membuat plot
27     plt.title(title) #membuat title pada plot
28     #plt.colorbar()
29     tick_marks = np.arange(len(classes)) #membuat marks pada
30     plot
31     plt.xticks(tick_marks, classes, rotation=90) #membuat
32     ticks pada x
33     plt.yticks(tick_marks, classes) #membuat ticks pada y
34
35     fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
36     thresh = cm.max() / 2. #variabel thresh menambil data max
37     pada cm kemudian dibagi 2
38
39
40
41 # In [22]:
42
43 birds = pd.read_csv("data/CUB_200_2011/classes.txt",
44                     sep='\s+', header=None, usecols=[1], names
45                     =[ 'birdname']) #membaca csv dengan ketentuan nama birdname
46 birds = birds[ 'birdname'] #nama birds dengan ketentuan
47         birdname
48 birds #menampilkan data birds
49
50
51 # In [23]:
52
53 import numpy as np #import library numpy sebagai np
54 np.set_printoptions(precision=2) #np sebagai variabel yang
55         membuat set precision=2
56 plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure
57         dengan ketentuan sizw 60,60 dan dpi 300
58 plot_confusion_matrix(cm, classes=birds, normalize=True) #data
59         cm dan clas birds dibuat sebagai plot
```

```
47 plt.show() #menampilkan hasil plot yang berbentuk grafik
```

Hasilnya adalah seperti dibawah ini :



**Gambar 3.157** Menampilkan hasil plot

6. Jalankan program klasifikasi SVM dan Decission Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Menunjukkan klarifikasi dengan decission tree menggunakan dataset yang sama dan akan memunculkan akurasi predksi.

```
1 # In[24]: Mencoba dengan metode Decission Tree dan SVM
2
3 from sklearn import tree #import library tree
4 clftree = tree.DecisionTreeClassifier() #clftree sebagai
5          variabel untuk decision tree
6 clftree.fit(df_train_att , df_train_label) #sebagai data
7          training
8 clftree.score(df_test_att , df_test_label) #sebagai data
9          testing
```

Hasilnya adalah seperti ini :

```
In [27]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[27]: 0.26135163674762407
```

**Gambar 3.158** Hasil Soal 6 - 1

- Menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi.

```

1 # In [25]:
2
3 from sklearn import svm #import library svm
4 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur
5         fungsi SVC
6 clfsvm.fit(df_train_att , df_train_label) #sebagai data
7         training
8 clfsvm.score(df_test_att , df_test_label) #sebagai data testing

```

Hasilnya adalah seperti ini :

```

In [28]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
Out[28]: 0.47729672650475186

```

**Gambar 3.159** Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

```

1 # In [26]: Pengecekan Cross Validation
2
3 from sklearn.model_selection import cross_val_score #import
4         cross_val_score
5 scores = cross_val_score(clf , df_train_att , df_train_label , cv
6         =5) #variabel scores sebagai variabel prediksi dari data
7         training
8 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
9         std() * 2)) #print data scores dengan ketentuan akurasi

```

Hasilnya adalah seperti ini :

```

In [29]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: scores
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.45 (+/- 0.03)

```

**Gambar 3.160** Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

```

1 # In [27]:
2
3 scorestree = cross_val_score(clftree , df_train_att ,
4         df_train_label , cv=5) #sebagai prediksi menggunakan scores
5         dan metode tree
6 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
7         scorestree.std() * 2)) #menampilkan dengan ketentuan yang
8         ada

```

Hasilnya adalah seperti ini :

```
[In [30]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.26 (+/- 0.01)
```

**Gambar 3.161** Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

```
[# In [28]:
1 scoresvm = cross_val_score(clfsvm, df_train_att,
2                             df_train_label, cv=5) #sebagai data training
3 print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(),
4                                         scoresvm.std() * 2)) #sebagai data testing dan output
5 akurasi
```

Hasilnya adalah seperti ini :

```
[In [31]: scoresvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoresvm.mean(), scoresvm.std() * 2))
Accuracy: 0.47 (+/- 0.03)
```

**Gambar 3.162** Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

```
[# In [29]: Pengamatan komponen informasi
1 max_features_opts = range(5, 50, 5) #max_features_opts sebagai
2     variabel untuk membuat range 5,50,5
3 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
4     sebagai variabel untuk membuat range 10,200,20
5 rf_params = np.empty((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=max_features
12            , n_estimators=n_estimators) #menampilkan variabel csf
13        scores = cross_val_score(clf, df.train_att,
14            df_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
18        rf_params[i,3] = scores.std() * 2 #index 3
19        i += 1 #dengan ketentuan i += 1
20        print("Max features: %d, num estimators: %d, accuracy:
21             %0.2f (+/- %0.2f)" % (max_features, n_estimators,
22             scores.mean(), scores.std() * 2))
```

```
17 #print hasil pengulangan yang sudah ditentukan
```

Hasilnya adalah seperti ini :

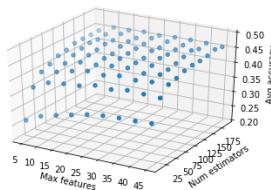
Max Features: 5, num estimators: 10, accuracy: 0.46 (+/- 0.02)	Max Features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 25, num estimators: 10, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 50, accuracy: 0.40 (+/- 0.03)	Max Features: 25, num estimators: 30, accuracy: 0.48 (+/- 0.02)
Max Features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.03)	Max Features: 25, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max Features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)	Max Features: 25, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max Features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max Features: 25, num estimators: 90, accuracy: 0.45 (+/- 0.03)
Max Features: 5, num estimators: 130, accuracy: 0.44 (+/- 0.03)	Max Features: 25, num estimators: 110, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max Features: 25, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)	Max Features: 25, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.02)	Max Features: 25, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max Features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.01)	Max Features: 30, num estimators: 10, accuracy: 0.45 (+/- 0.04)
Max Features: 10, num estimators: 30, accuracy: 0.39 (+/- 0.03)	Max Features: 30, num estimators: 30, accuracy: 0.48 (+/- 0.02)
Max Features: 10, num estimators: 50, accuracy: 0.39 (+/- 0.03)	Max Features: 30, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max Features: 10, num estimators: 70, accuracy: 0.41 (+/- 0.03)	Max Features: 30, num estimators: 70, accuracy: 0.45 (+/- 0.03)
Max Features: 10, num estimators: 90, accuracy: 0.42 (+/- 0.02)	Max Features: 30, num estimators: 90, accuracy: 0.45 (+/- 0.03)
Max Features: 10, num estimators: 110, accuracy: 0.42 (+/- 0.03)	Max Features: 30, num estimators: 110, accuracy: 0.46 (+/- 0.03)
Max Features: 10, num estimators: 130, accuracy: 0.43 (+/- 0.04)	Max Features: 30, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max Features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.04)	Max Features: 30, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.04)	Max Features: 30, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max Features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)	Max Features: 30, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 10, accuracy: 0.33 (+/- 0.02)	Max Features: 35, num estimators: 10, accuracy: 0.45 (+/- 0.04)
Max Features: 15, num estimators: 30, accuracy: 0.39 (+/- 0.02)	Max Features: 35, num estimators: 30, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 50, accuracy: 0.40 (+/- 0.03)	Max Features: 35, num estimators: 50, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.03)	Max Features: 35, num estimators: 70, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max Features: 35, num estimators: 90, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.04)	Max Features: 35, num estimators: 110, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 130, accuracy: 0.45 (+/- 0.04)	Max Features: 35, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.04)	Max Features: 35, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.04)	Max Features: 35, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max Features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.03)	Max Features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 10, accuracy: 0.33 (+/- 0.02)	Max Features: 40, num estimators: 10, accuracy: 0.45 (+/- 0.04)
Max Features: 20, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 40, num estimators: 30, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 40, num estimators: 50, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 70, accuracy: 0.43 (+/- 0.03)	Max Features: 40, num estimators: 70, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max Features: 40, num estimators: 90, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 110, accuracy: 0.44 (+/- 0.04)	Max Features: 40, num estimators: 110, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 130, accuracy: 0.45 (+/- 0.04)	Max Features: 40, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 150, accuracy: 0.45 (+/- 0.04)	Max Features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 170, accuracy: 0.45 (+/- 0.04)	Max Features: 40, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max Features: 20, num estimators: 190, accuracy: 0.45 (+/- 0.03)	Max Features: 40, num estimators: 190, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 10, accuracy: 0.33 (+/- 0.02)	Max Features: 45, num estimators: 10, accuracy: 0.45 (+/- 0.04)
Max Features: 25, num estimators: 30, accuracy: 0.40 (+/- 0.03)	Max Features: 45, num estimators: 30, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max Features: 45, num estimators: 50, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 70, accuracy: 0.43 (+/- 0.03)	Max Features: 45, num estimators: 70, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max Features: 45, num estimators: 90, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 110, accuracy: 0.44 (+/- 0.04)	Max Features: 45, num estimators: 110, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 130, accuracy: 0.45 (+/- 0.04)	Max Features: 45, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 150, accuracy: 0.45 (+/- 0.04)	Max Features: 45, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 170, accuracy: 0.45 (+/- 0.04)	Max Features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max Features: 25, num estimators: 190, accuracy: 0.45 (+/- 0.03)	Max Features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.03)

Gambar 3.163 Hasil Soal 8 - 1

- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```
1 # In [30]:
2
3 import matplotlib.pyplot as plt #import library matplotlib sebagai plt
4 from mpl_toolkits.mplot3d import Axes3D #import axes3D untuk menampilkan plot 3 dimensi
5 from matplotlib import cm #memanggil data cm yang sudah tersedia
6 fig = plt.figure() #hasil plot sebagai figure
7 fig.clf() #figure di ambil dari clf
8 ax = fig.gca(projection='3d') #ax sebagai projection 3d
9 x = rf_params[:,0] #x sebagai index 0
10 y = rf_params[:,1] #y sebagai index 1
11 z = rf_params[:,2] #z sebagai index 2
12 ax.scatter(x, y, z) #membuat plot scatter x y z
13 ax.set_zlim(0.2, 0.5) #set zlim dengan ketentuan yang ada
14 ax.set_xlabel('Max features') #memberi nama label x
15 ax.set_ylabel('Num estimators') #memberi nama label y
16 ax.set_zlabel('Avg accuracy') #memberi nama label z
17 plt.show() #print hasil plot yang sudah dibuat.
```

Hasilnya adalah seperti ini :



**Gambar 3.164** Hasil Soal 8 - 2

### 3.7.3 Penanganan Error

#### 1. ScreenShoot Error

```
[Errno 2] No such file or directory: '/home/jl/1174012/dataset_ml_attributes/images_attributes_labels.txt' does not exist:
> Fatal error(s) detected during image_attributes_labels.txt
```

**Gambar 3.165** FileNotFoundError

#### 2. Tuliskan Kode Error dan Jenis Error

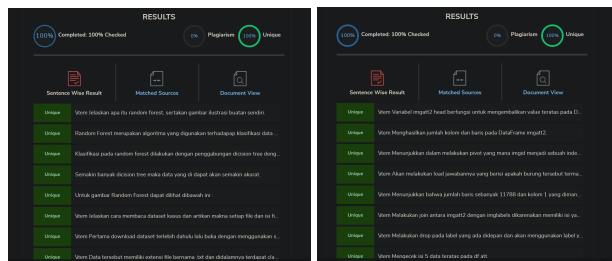
- FileNotFoundError

#### 3. Cara Penangan Error

- FileNotFoundError

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

### 3.7.4 Bukti Tidak Plagiat



**Gambar 3.166** Bukti Tidak Melakukan Plagiat Chapter 3

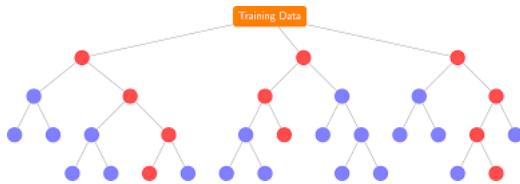
## 3.8 Damara Benedikta 1174012

### 3.8.1 Teori

#### 1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest adalah konstruk data yang diterapkan pada machine learning yang mengembangkan sejumlah besar pohon keputusan acak yang menganalisis sekumpulan variabel. Jenis algoritma ini membantu meningkatkan cara teknologi menganalisis data yang kompleks. Juga merupakan algoritma machine learning yang fleksibel, mudah digunakan, bahkan tanpa penyetelan hyperparameter, dengan hasil yang baik. Ini juga merupakan salah satu algoritma yang paling banyak digunakan, karena kesederhanaan dan faktanya dapat digunakan untuk tugas klasifikasi dan regresi.

Dibawah ini merupakan salah satu ilustrasi penggunaan Random Forest.



**Gambar 3.167** Random Forest

2. Jelaskan cara membaca dataset khusus dan artikan makna setiap file dan isi field masing masing file. Dataset adalah kumpulan data. Paling umum satu dataset sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, di mana setiap kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.
  - Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
  - Setelah itu, buat variabel baru "dataset" yang berisikan perintah untuk membaca file csv.
  - Memanggil Librari Panda untuk membaca dataset
  - Membuat variabel "Dataset" yang berisikan pdreadcsv untuk membaca dataset. Pada contoh ini menggunakan txt tapi tetap bisa membaca datasetnya, mengapa? Karena pada saat dijalankan librari panda secara otomatis akan mengubah data dalam bentuk text file ke format csv.
3. Jelaskan apa itu Cross Validation. Cross Validation adalah prosedur resampling yang digunakan untuk mengevaluasi model machine learning pada sampel data yang terbatas. Prosedur ini memiliki parameter tunggal yang disebut k yang mengacu pada jumlah grup tempat sampel data yang akan dibagi. Karena itu, prosedur ini sering disebut k-fold cross-validation. Proses penentuan apakah hasil numerik yang mengukur hubungan yang dihipotesiskan antar variabel, dapat diterima sebagai deskripsi data, dikenal sebagai Validationi. Umumnya, estimasi kesalahan untuk model dibuat setelah training, lebih dikenal sebagai evaluasi residu. Dalam proses ini, estimasi numerik dari perbedaan respons yang diprediksi dan yang asli dilakukan, juga disebut kesalahan training. Namun, ini

hanya memberi kita gambaran tentang seberapa baik model kita pada data yang digunakan untuk melatihnya. Sekarang mungkin bahwa model tersebut kurang cocok atau overfitting data. Jadi, masalah dengan teknik evaluasi ini adalah bahwa itu tidak memberikan indikasi seberapa baik pelajar akan menggeneralisasi ke set data independen / tidak terlihat. Model ini dikenal sebagai Cross Validation.

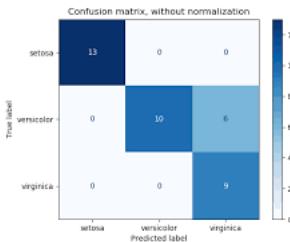
4. Jelaskan apa arti score 44 % pada random forest, 27 % pada decision tree dan 29 % dari SVM. Itu merupakan persentase keakurasi prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasi atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decision Tree hasil prediksinya yaitu 27% dan pada SVM 29% .
  5. Jelaskan bagaimana cara membaca confusion matrix dan contohnya memakai gambar atau ilustrasi sendiri. Perhitungan Confusion Matrix dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.
    - Import library 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 di plotting.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:28:30 2020
4
5 @author: Damara
6 """
7
8 import numpy as np
9 import matplotlib.pyplot as plt
10 import pandas as pd
11 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name=
12     'Actual')
13 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name=
14     'Predicted')
15 df_confusion = pd.crosstab(y_actu, y_pred)
16 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual' ,
17     ], colnames=[ 'Predicted' ], margins=True)
```

```

15 def plot_confusion_matrix(df_confusion , title='Confusion
16     matrix' , cmap=plt.cm.gray_r):
17     plt.matshow(df_confusion , cmap=cmap) # imshow
18     #plt.title(title)
19     plt.colorbar()
20     tick_marks = np.arange(len(df_confusion.columns))
21     plt.xticks(tick_marks , df.confusion.columns , rotation=45)
22     plt.yticks(tick_marks , df.confusion.index)
23     #plt.tight_layout()
24     plt.ylabel(df.confusion.index.name)
25     plt.xlabel(df.confusion.columns.name)
26 plot_confusion_matrix(df_confusion)
27 plt.show()

```

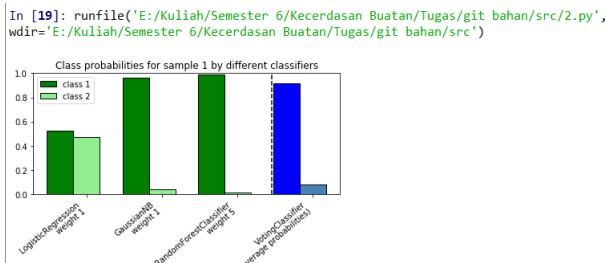


**Gambar 3.168** Confusion Matriks

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

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

- Untuk menggunakan Voting pada Random Forest dapat dilihat code berikut. Disini saya mengilustrasikan voting untuk berbagai macam algoritma terutama Random Forest.



**Gambar 3.169** Voting

### 3.8.2 Praktikum

#### 1. pandas

pada baris ke satu yaitu perintah mengimport library padas pada python atau anaconda kemudian di inisialisasi menjadi karakter. selanjutnya ada sebuah array yang berisi a b c d. selanjutnya penggunaan array tipe series dan yang terakhir perintah print untuk menampilkan data pada karakter.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:06:12 2020
4
5 @author: ACER
6 """
7
8 import pandas as pd
9 data = np.array(['a', 'b', 'c', 'd'])
10 karakter = pd.Series(data)
11 print(karakter)
```

```

In [20]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/3.py',
               wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')
0    a
1    b
2    c
3    d
dtype: object
```

**Gambar 3.170** hasil

#### 2. numpy

Arti tiap baris codingan pada aplikasi sederhana numpy adalah sebagai berikut : pada baris ke satu yaitu mengimport numpy yang di inisialisasi menjadi np kemudian pada baris selanjutnya berisikan arange yang berarti membuat data yang berisi 12 dan ada reshape yang berfungsi merubah bentuk dari satu baris menjadi 2 baris data. Lalu yang terakhir ada perintah untuk print yaitu menampilkan data dari dika.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 12:12:39 2020
4
5 @author: ACER
6 """
7
8 import numpy as np
9 tomy=np.arange(12).reshape(6,2)
10 print(tomy)
```

#### 3. matplotlib

Arti tiap baris aplikasi sederhana matplotlib pada baris ke satu yaitu memasukan library matplotlib.pyplot yang di definisikan menjadi plt kemudian plt.plot un-

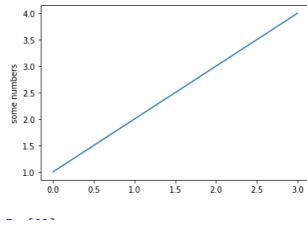
```
In [21]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/4.py',
wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')
[[ 0  1]
 [ 2  3]
 [ 4  5]
 [ 6  7]
 [ 8  9]
[10 11]]
```

**Gambar 3.171** hasil

tuk menentukan grafik yang akan dibuat. lalu membuat variabel y dengan nama some number yang terakhir untuk menampilkan data pada sebuah grafik.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:31:45 2020
4
5 @author: Damara
6 """
7
8 import matplotlib.pyplot as plt
9 plt.plot([1, 2, 3, 4])
10 plt.ylabel('some numbers')
11 plt.show()
```

```
In [22]: runfile('E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src/5.py',
wdir='E:/Kuliah/Semester 6/Kecerdasan Buatan/Tugas/git bahan/src')
```

**Gambar 3.172** hasil

#### 4. Random Forest

Arti tiap baris hasil codingan random forest pada baris pertama random forest di import dari sklearn dengan ketentuan yaitu Nilai default untuk parameter yang mengontrol ukuran pohon (mis. Max\_depth, min\_samples\_leaf, dll.) Mengarah ke pohon yang tumbuh besar dan tidak di-unsung yang berpotensi sangat besar pada beberapa set data. Untuk mengurangi konsumsi memori, kompleksitas dan ukuran pohon harus dikontrol dengan menetapkan nilai parameter tersebut.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:32:13 2020
4
5 @author: Damara
6 """
7
```

```

8 from sklearn.ensemble import RandomForestClassifier
9 from sklearn.datasets import make_classification
10 X, y = make_classification(n_samples=1000, n_features=4,
11                             n_informative=2, n_redundant=0,
12                             random_state=0, shuffle=False)
13 clf = RandomForestClassifier(max_depth=2, random_state=0)
14 clf.fit(X, y)
15 print(clf.feature_importances_)
16 print(clf.predict([[0, 0, 0, 0]]))

```

```

In [27]: X, y = make_classification(n_samples=1000, n_features=4,
...:                                         n_informative=2, n_redundant=0,
...:                                         random_state=0, shuffle=False)
...:                                         random_state=0)
...:                                         clf = RandomForestClassifier(max_depth=2, random_state=0)
...:                                         clf.fit(X, y)
Out[27]:
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
criterion='gini', max_depth=2, max_features='auto',
max_leaf_nodes=None, max_samples=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)

In [28]: print(clf.feature_importances_)
[0.14205973 0.76664038 0.0282433 0.06305659]

In [29]: print(clf.predict([[0, 0, 0, 0]]))
[1]

```

**Gambar 3.173** hasil

## 5. Confusion Matrix

arti codingan pada hasil tiap codingan confusion matrix pada baris pertama codingan tersebut mendeskripsikan atau mengimport confusion matrix dari sklearn kemudian dibuat variabel `y_true` untuk nilai target ground truth (benar). `y_pred` untuk Taksiran target seperti yang dikembalikan oleh classifier. lalu menampilkan kedua variabel.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:32:46 2020
4
5 @author: Damara
6 """
7
8 from sklearn.metrics import confusion_matrix
9 y_true = [2, 0, 2, 2, 0, 1]
10 y_pred = [0, 0, 2, 2, 0, 2]
11 confusion_matrix(y_true, y_pred)

```

## 6. SVM dan Decision Tree

Seperti pengklasifikasi lainnya, `DecisionTreeClassifier` mengambil input dua array: array `X`, jarang atau padat, dengan ukuran `n_samples`, `n_features` memegang sampel pelatihan, dan array `Y` dari nilai integer, ukuran `n_samples`. Atau, probabilitas setiap kelas dapat diprediksi. Seperti pengklasifikasi lainnya, `SVC`,

```
In [35]: from sklearn.metrics import confusion_matrix
... y_true = [2, 0, 2, 2, 0, 1]
... y_pred = [0, 0, 2, 2, 0, 2]
... confusion_matrix(y_true, y_pred)
Out[35]:
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
```

**Gambar 3.174** hasil

NuSVC dan LinearSVC mengambil input dua array: array X ukuran n\_samples, n\_features memegang sampel pelatihan, dan array y label kelas (string atau bilangan bulat), ukuran n\_samples:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:33:23 2020
4
5 @author: Damara
6 """
7
8 from sklearn import tree
9 X = [[0, 0], [1, 1]]
10 Y = [0, 1]
11 clf = tree.DecisionTreeClassifier()
12 clf = clf.fit(X, Y)
13 clf.predict([[2., 2.]])
14
15 from sklearn import svm
16 X = [[0, 0], [1, 1]]
17 y = [0, 1]
18 clf = svm.SVC()
19 clf.fit(X, y)
20 clf.predict([[2., 2.]])
```

```
In [40]: from sklearn import svm
... X = [[0, 0], [1, 1]]
... y = [0, 1]
... clf = svm.SVC()
... clf.fit(X, y)
... clf.predict([[2., 2.]])
Out[40]: array([1])

In [41]: from sklearn import tree
... X = [[0, 0], [1, 1]]
... Y = [0, 1]
... clf = tree.DecisionTreeClassifier()
... clf = clf.fit(X, Y)
... clf.predict([[2., 2.]])
Out[41]: array([1])
```

**Gambar 3.175** hasil

## 7. Cross Validation

digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut untuk lebih jelasnya dapat di lihat pada gambar pada codingan tersebut pada baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya

jutnya mengisi nilai skor dengan nilai pada variabel lontong setelah hal tersebut dilakukan kemudian data tersebut di eksekusi. berikut merupakan hasil dari code tersebut dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:33:59 2020
4
5 @author: Damara
6 """
7
8 from sklearn.model_selection import cross_val_score
9 scores = cross_val_score(lontong, pecel_att, pecel_pass, cv=5)
10 # show average score and +/- two standard deviations away
11 # (covering 95% of scores)
12 print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std()
13     () * 2))

```

```
In [61]: runfile('D:/SEMESTER 6/wert/9.py', wdir='D:/SEMESTER 6/wert')
Accuracy: 0.59 (+/- 0.05)
```

**Gambar 3.176** hasil

8. program pengamatan arti dari hasil program pengamatan. perogram pengamatan dapat mengamati dari 3 aspek diatas yaitu svm, random, dan decision tree. Yang memiliki variabel X Y Z dan di tampilkan dalam bentuk grafik.

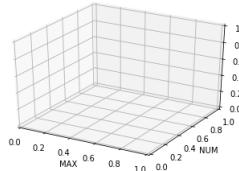
```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 13:34:23 2020
4
5 @author: Damara
6 """
7
8 import matplotlib.pyplot as plt
9 from mpl_toolkits.mplot3d import Axes3D
10 from matplotlib import cm
11 fig = plt.figure()
12 fig.clf()
13 ax = fig.gca(projection='3d')
14 plt.xlabel('MAX')
15 plt.ylabel('NUM')
16 plt.zlabel('AVG')
17 ax.scatter(x, y, z)
18 ax.set_zlim(0.2, 0.5)
19 ax.set_xlabel('Max features')
20 ax.set_ylabel('Num estimator')
21 ax.set_zlabel('Avg accuracy')
22 plt.show()

```

### 3.8.3 Penanganan Error

Screenshot error



**Gambar 3.177** hasil

### 1. Untuk gambar screenshot error

```

File "<ipython-input-60-d14a3944647a>", line 1, in <module>
    runfile('D:/SEMESTER 6/wert/1/5.py', wdir='D:/SEMESTER 6/wert/1')

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 827, in runfile
    execfile(filename, namespace)

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 110, in execfile
    exec(compile(f.read(), filename, 'exec'), namespace)

File "D:/SEMESTER 6/wert/1/5.py", line 10
    plt.ylabel(some numbers)
                           ^
SyntaxError: invalid syntax

```

**Gambar 3.178** hasil

### Code Errornya

```

import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel(some numbers)
plt.show()

```

pada kode ylabel memiliki nama atau isi some numbers tetapi pada tipe data tertentu harus di awali dan diakhiri dengan tanda petik 2 atau 1. Pada kodingnya hanya kurang tanda petik 1.

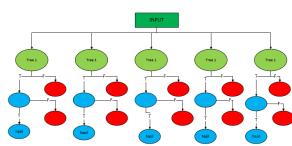
## 3.9 Evietania - 1174051

### 3.9.1 Teori

#### 1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest atau hutan acak yaitu kumpulan dari pohon-pohon keputusan yang digunakan untuk membaca objek tertentu yang telah di sepakati untuk dibaca dalam AI. pohon-pohon keputusan tersebut akan memunculkan hasil-hasil yang akan disimpulkan oleh random forest. pembagian jumlah data yang dimasukan kedalam decision tree pada random forest akan di bagi sama rata

sesuai codingan atau ketentuan tertentu yang di sepakati. misalkan data yang akan digunakan sebanyak 314 jika dalam satu decision tree di putuskan untuk memiliki 50 data maka pada satu random forest akan terdapat enam atau tujuh decision tree.



**Gambar 3.179** contoh binari calssification

2. Jelaskan cara membaca dataset khusus dan artikan makna setiap file dan isi field masing masing file. langkah pertama download terlebih dahulu dataset nya kemudian buka menggunakan spyder bawaan anaconda untuk mengetahui isi dari dataset tersebut. biasanya data tersebut berisi databerekstensi .txt yang di dalamnya terdapat class dari field atau data data yang ada data tersebut. contoh pada data burung ada field index dan angka, index biasanya berisi angka, angka angka tersebut memiliki makna yaitu pengganti nama atau jenis dari burung tersebut sedangkan pada field yang berisi nilai 0 dan 1 berarti menyatakan atau maknanya yaitu memberikan nilai ya dan tidak nilai tersebut di ubah menjadi angka nol dan satu karna data pada field tersebut harus berisi nilai boolean atau pilihan ya dan tidak di karenakan komputer susah membaca nilai dan tidak maka di ubahlah menjadi 0 dan 1 dengan 0 bernilai tidak dan 1 bernilai ya.
3. Jelaskan apa itu Cross Validation. Cross Validation merupakan cara untuk mengevaluasi hasil dari sebuah metode yang telah digunakan dengan cara membagi dua bagian dari dataset menjadi data training dan data testing kemudian data tersebut diolah hingga muncul tingkat akurasi dari metode yang digunakan contoh pada metode random forest dataset nya di bagi menjadi dua menjadi data training dan data testing kemudian data tersebut di olah oleh mesin untuk melihat tingkat akurasinya maka akan muncul misalkan akurasi kebenaran sebesar 44 % begitu pula dengan menggunakan metode-metode yang lain seperti decision tree dan SVM.
4. Jelaskan apa arti score 44 % pada random forest, 27 % pada decision tree dan 29 % dari SVM. maksud dari score 44 % tersebut yaitu nilai ketepatan atau kebenaran atau bisa disebut hasil dari random forest misalkan dengan metode random forest mesin membaca objek burung, mesin tersebut bisa menyatakan jenis burung tersebut dengan akurasi kebenaran 44 %. sedangkan pada metode decision tree yaitu 27 % yang berarti menunjukkan bahwa tingkat akurasi ketepatan mesin jika mengerjakan sesuatu atau menyatakan keputusan dengan metode decision tree maka nilai kebenarannya bernilai 27 %. sedangkan dengan menggunakan metode SVM menunjukan hasil 29 % yang berarti nilai ketepatan atau kebenaran dalam memecahkan masalah menggunakan metode SVM ini sebesar

29 % . maka dari itu dapat di simpulkan bahwa dengan menggunakan metode random forest mesin dapat memecahkan masalah lebih akurat dibandingkan dengan menggunakan decision tree dan SVM.

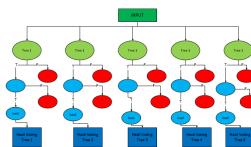
- Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri. cara membaca confusio matrix dengan cara memasukan para meter nilai yang ada pada datasets contoh pada dataset ter-dapat class yang disandingkan dengan nama burung untuk di normalisasi maka akan menunjukkan nilai matrix yang mendekati nilai benar dalam bentuk angka misalkan 0,5 0,2 dan seterusnya mendekati nilai satu. di karenakan susahnya membaca nilai angka maka sering di ubah menjadi bentuk grafik.

Burung A	Blue	White	White	White	White	White
Burung B	Blue	Blue	Blue	Blue	Blue	Blue
Burung C	Blue	Blue	Blue	Blue	Blue	Blue
Burung D	Blue	Blue	Blue	Blue	Blue	Blue
Burung E	Blue	Blue	Blue	Blue	Blue	Blue
Burung F	Blue	Blue	Blue	Blue	Blue	Blue
Burung G	Blue	Blue	Blue	Blue	Blue	Blue

**Gambar 3.180** contoh binari calssification

- Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

Voting merupakan data hasil dari decision tree yang terdapat pada random forest. Dimana hasil data tersebut di gunakan sebagai acuan untuk hasil dari random forest. sebagai contoh misalkan pada satu random forest terdapat enam decision tree untuk menentukan jenis pekerjaan orang, pada decision tree ke satu menyimpulkan bahwa pekerjaanya yaitu dosen , pada decision tree ke dua yaitu dosen kemudian pada decision tiga dosen , pada decision tree ke empat yaitu pekerja kantoran, pada decision tree ke lima yaitu pekerja kantoran dan pada decision tree ke enam yaitu dosen. maka pada random forest dapat menyimpulkan hasilnya yaitu dosen.



**Gambar 3.181** contoh binari calssification

### 3.9.2 Praktikum

- pandas

pada baris ke satu yaitu perintah mengimport library padas pada python atau anaconda kemudian di inisialisasikan menjadi kue. selanjutnya pada baris ke 3 terdapat nama variabel yaitu nama\_kue\_tradisional = yang di dalamnya

terdapat tiga nama field yakni Name Kue, harga satuan dan terbilang kemudian pada baris ke tujuh terdapat variabel baru bernama Data\_kue = kemudian didalamnya mendeskripsikan kue berdasarkan tipe DataFrame yang berisi variabel nama\_kue\_tradisional selanjutnya data tersebut di cetak pada console dengan perintah print (Data\_kue).

```

1 import pandas as kue
2 nama_kue = { 'Nama Kue' : [ 'Cuhcur', 'Putri Noong', 'Bugis', 'Papais', 'Ali-Ali'],
3   'Harga Satuan' : [2000,5000,1500,2500,1000], 'Terbilang' : [ 'Dua
4   Ribu Rupiah', 'Lima Ribu Rupiah',
5   'Seribu Lima Ratus Rupiah', 'Dua Ribu Limaratus Rupiah', 'Seribu
6   Rupiah']}
7 Data_kue = kue . DataFrame (nama_kue)
8 print (Data_kue)
```

```
In [1]: runfile('D:/kuliah/smt 6/kecerdasan/33/2,1.py', wdir='D:/kuliah/smt 6/kecerdasan/
33')
      Nama Kue    Harga Satuan        Terbilang
0       Cuhcur        2000      Dua Ribu Rupiah
1     Putri Noong        5000      Lima Ribu Rupiah
2        Bugis        1500  Seribu Lima Ratus Rupiah
3      Papais        2500  Dua Ribu Limaratus Rupiah
4      Ali-Ali        1000      Seribu Rupiah
```

**Gambar 3.182** hasil

## 2. numpy

Arti tiap baris codingan pada aplikasi sederhana numpy adalah sebagai berikut : pada baris ke satu yaitu mengimport numpy yang di inisialisasi menjadi np kemudian pada baris ke tiga dibut variabel ali yang berisi numpy bertipekan arrange 6 yang berarti berisi nilai array dari 0 sampai 5 kemudian pada baris ke empat di cetak hasilnya dengan memasukan perintah print (ali) selanjutnya yaitu membuat nilai array tiga dimensi pada baris ke tujuh dengan cara membuat variabel botak yang berisi rank nilainya kemudian dimensinya yaitu 4 3 3 kemudian variabel tersebut di print. selanjutnya pada baris ke 12 dibuat variabel nilai\_array\_1 dengan isian nilai array 1 2 3 4 kemudian pada baris ke 13 di buat variabe nilai\_array\_2 dengan nilai array 20 30 40 dan 50 selanjutnya pada baris ke 14 dibut nilai variabel Nilai\_array\_3 dengan rank 4 yang berarti berisi nilai dari 0 sampai 3 setelah itu di buat variabel hasil dimana isinya yaitu penjumlahan nilai\_array\_1+Nilai\_array\_3+Nilai\_array\_3 setelah itu nilai\_array\_1 , Nilai\_array\_3, dan Hasil di prin untuk melihat nilai dari array tersebut.

```

1 import numpy as np
2
3 epi = np.arange(6)
4 print(epi)
5
6 #array 3dimensi
7 epik = np.arange(36).reshape(4,3,3)
8 print(epik)
9
10 #penjumlahan array
```

```

11 nilai_array_1 = np.array([1,2,3,4])
12 nilai_array_2 = np.array([20,30,40,50])
13 nilai_array_3 = np.arange(4)
14 Hasil = nilai_array_1+nilai_array_3=nilai_array_3
15 print(nilai_array_1)
16 print(nilai_array_3)
17 print(Hasil)

```

```

In [3]: runfile('D:/kuliah/smt 6/kecerdasan/33/2,2.py', wdir='D:/kuliah/smt 6/kecerdasan/
33')
[[0 1 2 3 4 5]
 [[ 0  1  2]
 [ 3  4  5]
 [ 6  7  8]]

 [[ 9 10 11]
 [12 13 14]
 [15 16 17]]

 [[18 19 20]
 [21 22 23]
 [24 25 26]]

 [[27 28 29]
 [30 31 32]
 [33 34 35]]]
[[1 2 3 4]
[0 1 2 3]
[ 1 4 7 10]

In [4]: |

```

**Gambar 3.183** hasil

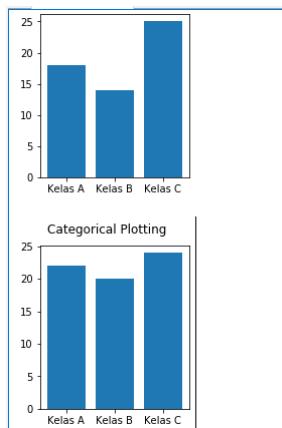
### 3. matplotlib

Arti tiap baris aplikasi sederhana matplotlib pada baris ke satu yaitu memasukan library matplotlib.pyplot yang di definisikan menjadi plt kemudian membuat variabel kelas\_ti3 pada baris ke tiga yang berisi label setelah itu di buat variabel jumlah\_mhs3 pada baris ke empat yang berisi nilai dari setiap label tersebut. begitu juga pada baris ke emam dan ke tujuh kemudian pada baris ke sembilan matplotlib mendefinisikan gambar dengan ukurannya dan pada baris ke 10 di dekralasikan subplot setelah itu pada baris ke 11 matplotlib mendefinisikan jenis grafik yang digunakan dan dimasukan variabel kelas dan jumlah\_mhs. begitujuga oada baris ke 13 14 dan 15 setelah itu di buat title pada baris ke 17 dan matplotlib di show untuk mendapatkan hasil dari grafiknya.

```

1 import matplotlib.pyplot as plt
2
3 kelas_ti3 = ['Kelas A', 'Kelas B', 'Kelas C']
4 jumlah_mhs3 = [18, 14, 25]
5
6 kelas_ti2 = ['Kelas A', 'Kelas B', 'Kelas C']
7 jumlah_mhs2 = [22, 20, 24]
8
9 plt.figure(1, figsize=(9,3))
10 plt.subplot(131)
11 plt.bar(kelas_ti3, jumlah_mhs3)
12 plt.figure(2, figsize=(9,3))
13 plt.subplot(132)
14 plt.bar(kelas_ti2, jumlah_mhs2)
15 plt.suptitle('Categorical Plotting')
16 plt.show()

```



Gambar 3.184 hasil

#### 4. Random Forest

Arti tiap baris hasil codingan random forest pada baris pertama random forest di import dari sklearn dengan ketentuan yaitu maksimal isi dari decision tree berisi 50 data dengan keadaan random dan dengan estimators 100 data ini berada dalam variabel clf kemudian setelah itu variabel clf di running berdasarkan data training dan data label yang telah di definisikan jumlahnya. kemudian variabel clf di running berdasarkan data training paling atas untuk memunculkan hasil data training lima paling atas. setelah itu data tersebut di di running scorenya untuk melihat tingkat akurasi yang dia kerjakan maka tingkat akurasi yang dihasilkan berada pada kisaran 0,437 atau kisaran 43 %.

```

1 from sklearn.ensemble import RandomForestClassifier
2 clf = RandomForestClassifier(max_features=50, random_state=0,
   n_estimators=100)
3 clf.fit(df_train_att, df_train_label)
4 print(clf.predict(df_train_att.head()))
5 clf.score(df_test_att, df_test_label)

```

#### 5. Confusion Matrix

arti codingan pada hasil tiap codingan confusion matrix pada baris pertama codingan tersebut mendeskripsikan atau mengimport confusion matrix dari sklearn kemudian dibuat variabel pred\_labels dengan di isikan clf prdic df\_test\_att setelah itu membuat variabel cm yang isinya terdapat data yang di buat confusion matrix berdasarkan data test setelah variabel cm akan di running yang mana akan menghasilkan gambar berupa matrix matrix tersebut berisi nilai nilai kebenaran yang mendekati nilai benar atau mutlak nilai benar.

```

1 from sklearn.metrics import confusion_matrix
2 pred_labels = clf.predict(df_test_att)
3 cm = confusion_matrix(df_test_label, pred_labels)
4 cm

```

```

Terminal: Local + 
>>>
>>> from sklearn.ensemble import RandomForestClassifier
>>> clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
>>> clf.fit(df_train_att, df_train_label)
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                      criterion='gini', max_depth=None, max_features=50,
                      max_leaf_nodes=None, max_samples=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)
>>> print(clf.predict(df_train_att.head()))
[[0 0 ... 0 0 1]
 [0 0 ... 0 0 1]
 [0 0 ... 0 0 1]
 [0 0 ... 0 1 0]
 [0 0 ... 0 0 0]]
>>> clf.score(df_test_att, df_test_label)
0.009503695881731784
>>> []

```

**Gambar 3.185** hasil

```

>>> from sklearn.metrics import confusion_matrix
>>> pred_labels = clf.predict(df_test_att)
>>> cm = confusion_matrix(df_test_label, pred_labels)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "C:\Users\najib\AppData\Local\Programs\Python\Python37\lib\site-packages\sklearn\metrics\_classification.py", line 270, in confusion_matrix
    raise ValueError("%s is not supported" % y_type)
ValueError: multilabel-indicator is not supported
>>> cm
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'cm' is not defined

```

**Gambar 3.186** hasil

## 6. SVM dan Decision Tree

Arti dari setiap baris hasil codingan decision tree dan SVM pada tree masukan terlebih dahulu library tree setelah itu buat variabel clftree yang berisi decision tree setelah itu masukan nilai data training dan label yang telah di deklarasikan tadi setelah itu running variabel tersebut untuk mendapatkan score 0,266 kisaran 26 sampai 27 % akurasinya kemudian pada svm juga hampir sama masukan terlebih dahulu librarynya setelah itu buat variabel clfsvm yang berarti berisi nilai data training dan data label dan pendeklarasian svm itu sendiri setelah itu di running untuk mendapatkan nilai akurasinya atau score sebesar 0,283 atau dalam kisaran 23 %.

```

1 from sklearn import tree
2 clftree = tree.DecisionTreeClassifier()
3 clftree .fit (df_train_att , df_train_label)
4 clftree .score (df_test_att , df_test_label)
5
6 from sklearn import svm

```

```

7 clfsvm = svm.SVC()
8 clfsvm.fit(df_train_att, df_train_label)
9 clfsvm.score(df_test_att, df_test_label)

```

```

>>> from sklearn import tree
>>> clftree = tree.DecisionTreeClassifier()
>>> clftree.fit(df_train_att, df_train_label)
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')
>>> clftree.score(df_test_att, df_test_label)
0.0008044333214158026
>>> █

```

**Gambar 3.187** hasil

## 7. Cross Validation

arti dari setiap baris hasil cross validation pada gambar?? tersebut diperlihatkan codingan error dikarenakan data training terlalu besar maka untuk mengatasinya halini dapat dilihat pada sub bab penanganan error

```

1 from sklearn.model_selection import cross_val_score
2 scores = cross_val_score(clf, df.train_att, df.train_label, cv=5)
3 print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std()
                                         () * 2))

```

```

>>> from sklearn.model_selection import cross_val_score
>>> scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
>>> print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.00 (+/- 0.00)
>>> █

```

**Gambar 3.188** hasil

8. program pengamatan arti dari hasil program pengamatan. perogram pengamatan ini menggunakan library matplotlib supaya hasil dari presentase hasil random forest, svm dan decision tree dapat di bandingkan dengan membuat variabel X Y Z kemudian memberikan label untuk setiap dimensinya untuk lebih jelas dapat dilihat gambar ?? yang menunjukan hasil perbandingan presentase dari tiga metode tersebut.

```

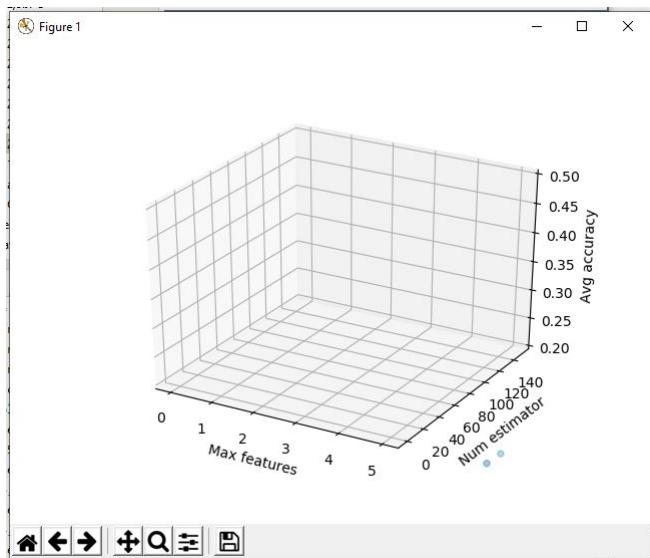
1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 from matplotlib import cm
4 fig = plt.figure()
5 fig.clf()
6 ax = fig.gca(projection='3d')
7 x = rf_params[:,0]
8 y = rf_params[:,1]
9 z = rf_params[:,2]
10 ax.scatter(x, y, z)

```

```

11 ax.set_zlim(0.2, 0.5)
12 ax.set_xlabel('Max features')
13 ax.set_ylabel('Num estimator')
14 ax.set_zlabel('Avg accuracy')
15 plt.show()

```



**Gambar 3.189** hasil

### 3.9.3 Penanganan Error / cokro

Screenshot error

- Untuk gambar screenshot error

Code Errornya

- kode error pada screenshot ke satu yaitu dikarenakan `clfsvm.fit(df_train_att, df_train_label)` dikarenakan data trainingnya terlalu besar sehingga komputernya error.
- untuk kode error pada screen shoot ke 2 sampai ke 4 dikarenakan pada kode berikut `scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)` `scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)` dan `scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)` hal ini di karenakan data trainingterlalu besar sehingga berdampak pada komputer sehingga library dari python tidak mampu mengolah data dan hasilnya menjadi error.

Solusi Untuk mengatasi Error

```
In [48]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Traceback (most recent call last):

File "<ipython-input-48-d7a7153ce4e9>", line 1, in <module>
    scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 402, in cross_val_score
    error_score=error_score)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 240, in cross_validate
    for train, test in cv.split(X, y, groups))

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 917, in __call__
    if self.dispatch_one_batch(iterator):

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 759, in dispatch_one_batch
    self._dispatch(tasks)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 716, in _dispatch
    job = self._backend.apply_async(batch, callback=cb)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\_parallel_backends.py", line 182, in apply_async
    result = ImmediateResult(func)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\_parallel_backends.py", line 549, in __init__
    self.results = batch()

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 225, in __call__
    for func, args, kwargs in self.items]

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 225, in <listcomp>
```

**Gambar 3.190** hasil

1. solusinya untuk yang ke satu yaitu dengan cara merestart spyder atau mematikannya kemudian nyalakan kembali lalu jalankan code yang error tersebut di CMD cika dalam python CMD jalam maka bisa di running. setelah itu buka kembali spyder dan jalankan codingan dari awal hingga pada bagian SVM tunggu sebenar sampai muncul nilai akurasinya.
2. solusi untuk mengatasi error tersebut yaitu dengan cara merubah bobot data pada data training.

```
df = imgatt2.join(imglabels)
df = df.sample(frac=1)
df_att = df.iloc[:, :312]
df_label = df.iloc[:, :312]
df_train_att = df_att[:600]
df_train_label = df_label[:600]
df_test_att = df_att[600:]
df_test_label = df_label[600:]
```

**Gambar 3.191** hasil

## BAB 4

---

# CHAPTER 4

---

### 4.1 1174006 - Kadek Diva Krishna Murti

Lore ipsum dolor sit amet, consectetur adipiscing elit.

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

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



**Gambar 5.1** Kecerdasan Buatan.

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

#### 5.1.1 Teori

#### 5.1.2 Praktek

#### 5.1.3 Penanganan Error

#### 5.1.4 Bukti Tidak Plagiat



**Gambar 5.2** Kecerdasan Buatan.

## BAB 6

---

# CHAPTER 6

---

### 6.1 1174006 - Kadek Diva Krishna Murti

Lore ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak ,  
2   title={Colenak: GPS tracking model for post-stroke rehabilitation  
3   program using AES-CBC URL encryption and QR-Code},  
4   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and  
5   Hasanudin, Trisna Irmayadi},  
6   booktitle={Information Technology, Information Systems and  
7   Electrical Engineering (ICITISEE), 2017 2nd International  
8   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 adipiscing elit.

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