

# Tugas

# Kecerdasan Buatan



Faisal Abdullah

1194014

Applied Bachelor of Informatics Engineering

Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering

*Politeknik Pos Indonesia*

Bandung 2022

‘Jika Kamu tidak dapat menahan lelahnya belajar,  
Maka kamu harus sanggup menahan perihnya Kebodohan.’  
Imam Syafi’i

## **Acknowledgements**

Pertama-tama kami panjatkan puji dan syukur kepada Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga Buku Pedoman Tingkat Akhir ini dapat diselesaikan.

## **Abstract**

Buku Pedoman ini dibuat dengan tujuan memberikan acuan, bagi mahasiswa Tingkat Akhir dan dosen Pembimbing. Pada intinya buku ini menjelaskan secara lengkap tentang Standar pengerjaan Intership dan Tugas Akhir di Program Studi D4 Teknik Informatika, dan juga mengatur mekanisme, teknik penulisan, serta penilaiannya. Dengan demikian diharapkan semua pihak yang terlibat dalam aktivitas Bimbingan Mahasiswa Tingkat Akhir berjalan lancar dan sesuai dengan standar.

# Contents

<b>1</b>	<b>Mengenal Kecerdasan Buatan dan Scikit-Learn</b>	<b>1</b>
1.1	Teori . . . . .	1
1.2	Instalasi . . . . .	2
1.3	Penanganan Error . . . . .	5
<b>2</b>	<b>Membangun Model Prediksi</b>	<b>7</b>
2.1	Teori . . . . .	7
2.2	scikit-learn . . . . .	8
2.3	Penanganan Error . . . . .	14
<b>3</b>	<b>Prediksi dengan Random Forest</b>	<b>15</b>
3.1	Teori . . . . .	15
3.1.1	Random Forest . . . . .	16
3.1.2	Confusion Matrix . . . . .	18
3.1.3	Mencoba dengan metode Decission Tree dan SVM . . . . .	19
3.1.4	Pengecekan Cross Validation . . . . .	20
3.1.5	Pengamatan komponen informasi . . . . .	20
3.2	Soal Teori . . . . .	21
3.3	Praktek Program . . . . .	21
3.4	Penanganan Error . . . . .	27
3.5	Presentasi Tugas . . . . .	28

# List of Figures

2.1	Ilutراس K-fold cross validation . . . . .	8
2.2	Import . . . . .	9
2.3	Source Code Task 1 . . . . .	9
2.4	Hasil Task . . . . .	9
2.5	Source Code Task 2 . . . . .	10
2.6	Hasil Task 2 . . . . .	10
2.7	Source Code Task 3 . . . . .	10
2.8	Hasil Task 3 . . . . .	10
2.9	Source Code Task 4 . . . . .	10
2.10	Hasil Task 4 . . . . .	10
2.11	Source Code Task 5 . . . . .	11
2.12	Hasil Task 5 . . . . .	11
2.13	Source Code Task 6 . . . . .	11
2.14	Hasil Task 6 . . . . .	11
2.15	Source Code Task 7 . . . . .	12
2.16	Hasil Task 7 . . . . .	12
2.17	Hasil Task 7 . . . . .	12
2.18	Source Code Task 8 . . . . .	12
2.19	Hasil Task 8 . . . . .	12
2.20	Source Code Task 9 . . . . .	12
2.21	Hasil Task 9 . . . . .	12
2.22	Source Code Task 10 . . . . .	13
2.23	Hasil Task 10 . . . . .	13
2.24	Source Code Task 11 . . . . .	13
2.25	Hasil Task 11 . . . . .	13
2.26	Source Code Task 12 . . . . .	14
2.27	Hasil Task 12 . . . . .	14
3.1	Hasil Pandas Sederhana . . . . .	22

3.2	Hasil Numpy Sederhana . . . . .	22
3.3	Hasil Numpy Sederhana . . . . .	23
3.4	Hasil Random Forest . . . . .	23
3.5	Hasil ConfusionMatrix . . . . .	24
3.6	Hasil Klasifikasi DT . . . . .	25
3.7	Hasil Cross Validation Random Forest . . . . .	26
3.8	Hasil Cross Validation Klasifikasi DT . . . . .	26
3.9	Hasil Cross Validation SVM . . . . .	26
3.10	Hasil pengamatan komponen informasi . . . . .	27
3.11	Hasil Plot Komponen Informasi . . . . .	27

# Chapter 1

## Mengenal Kecerdasan Buatan dan Scikit-Learn

### 1.1 Teori

Praktek teori penunjang yang dikerjakan :

1. Definisi, Sejarah dan perkembangan Kecerdasan Buatan *Artificial Intelligence*.  
*Artificial Intelligence* adalah ilmu yang ada pada bidang komputer yang memungkinkan suatu sistem untuk menyelesaikan permasalahan.  
Pada akhir tahun 1955 AI pertama muncul berkat Newell dan Simon dengan adanya perkembangan perkembangan *The Logic Theorist* .Pada 1956, istilah AI pertama kali diciptakan di Darmouth College ketika menyelenggarakan konferensi dengan nama *The Dartmouth summer research project on artificial intelligence*. konferensi tersebut diselenggarakan untuk memancing para ahli.
2. Definisi supervised learning, klasifikasi, regresi dan unsupervised learning. Data set, training set dan testing set.
  - Supervised Learning  
*Supervised Learning* sebuah pembelajaran yang ditentukan berdasarkan penggunaan traning set yang berlabel.
  - Klasifikasi  
Klasifikasi adalah mengidentifikasi suatu data menjadi sebuah bagian dari kelas berdasarkan label.
  - Regresi



Regresi adalah mendefinisikan relasi antara dua variable maupun lebih seperti variable terikat dan variabel bebas untuk melihat selisih nilai prediksi dengan nilai real.

- Data set, Training set dan Testing set

Data set adalah kumpulan data. Kemudian training set adalah data set yang berfungsi melatih suatu algoritma untuk mencapai suatu tujuan, dan testing set yaitu data set yang digunakan untuk mengetahui akurasi dari algoritma yang sudah di latih sebelumnya.

- Unsupervised Learning

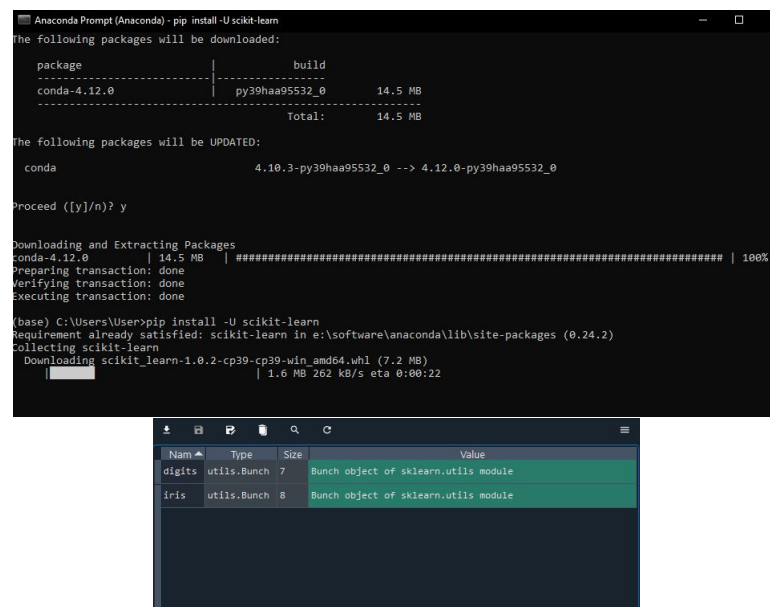
*Unsupervised Learning* sebuah pembelajaran yang ditentukan berdasarkan penggunaan training set yang tidak berlabel.

## 1.2 Instalasi

1. Instalasi library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer.

gunakan "pip install -U scikit-learn"

<https://youtu.be/xvX7Pye2Npw>



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

```

1 #-*- coding: utf-8 -*-
2 """
3 Spyder Editor
4 This is a temporary script file.
5 """
6
7
8
9
10
11 from sklearn import datasets #import class datasets dari scikit library
12
13 iris = datasets.load_iris() #membuat variable untuk load dataset iris
14 digits = datasets.load_digits() #membuat variable digits untuk load dataset digits
15 print(digits.data) #untuk menampilkan sampel digit
16 digits.target # untuk menampilkan target dari sampel digit
17 digits.images[0] #untuk menampilkan bentuk dari sampel

```

```

In [13]: print(digits.data) #untuk menampilkan sampel digit
Out[13]: array([[ 0.,  0.,  5., ...,  0.,  0.],
 [ 0.,  0.,  0., ..., 10.,  0.],
 [ 0.,  0.,  0., ..., 16.,  0.],
 ...,
 [ 0.,  0.,  1., ...,  0.,  0.],
 [ 0.,  0.,  2., ..., 12.,  0.],
 [ 0.,  0., 10., ..., 12.,  1.],
 ...,
 [ 0.,  0.,  0., ...,  0.,  0.]])

In [14]: digits.target # untuk menampilkan target dari sampel digit
Out[14]: array([0, 1, 2, ..., 0, 0, 0])

In [15]: digits.images[0]
Out[15]: array([[ 0.,  0.,  5., 13.,  0.,  1.,  0.,  0.],
 [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
 [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
 [ 0.,  0., 12.,  0.,  0.,  8.,  8.,  0.],
 [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
 [ 0.,  0., 11.,  0.,  0., 12.,  7.,  0.],
 [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
 [ 0.,  0.,  0., 13., 10.,  0.,  0.,  0.]])

In [16]:

```

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

```

1 """
2
3 from sklearn import svm, datasets
4
5 digits = datasets.load_digits()
6
7
8
9
10
11 clf = svm.SVC(gamma=0.001, C=100.)
12 # classifier clf dengan isi library svm dan class SVC
13 # menetapkan nilai gamma secara manual
14
15 clf.fit(digits.data[:-1], digits.target[:-1])
16 #method fit digunakan untuk training
17
18 clf.predict(digits.data[-1:])
19 #method predict digunakan untuk memprediksi value baru

```

```

In [6]: clf.fit(digits.data[:-1], digits.target[:-1])
Out[6]: SVC(C=100.0, gamma=0.001)

In [7]: clf.predict(digits.data[-1:])
Out[7]: array([8])

```

4. Mencoba Model persistence, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris. menggunakan 2 cara yaitu menggunakan pickle atau menggunakan joblib



```

19 X = transformer.fit_transform(X)
20 print(X.dtype)
21
22 clf = svm.SVC()
23 clf.fit(iris.data, iris.target)
24 print(clf.fit(iris.data, iris.target))
25
26 list(clf.predict(iris.data[:3]))
27
28 print((clf.predict(iris.data[:3])))
29
30 clf.fit(iris.data, iris.target_names[iris.target])
31 print(clf.fit(iris.data, iris.target_names[iris.target]))
32
33 list(clf.predict(iris.data[:3]))
34 print(list(clf.predict(iris.data[:3])))
35
36 #refitting and updating paramater
37 X, y = load_iris(return_X_y=True)
38 clf = svm.SVC()
39
40 clf.set_params(kernel='linear').fit(X, y)
41 print(clf.set_params(kernel='linear').fit(X, y))
42
43 clf.predict(X[:5])
44 print(clf.predict(X[:5]))
45
46 #Multiclass vs multilabel fitting
47 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
48 y = [0, 0, 1, 1, 2]
49
50 classif = OneVsRestClassifier(estimator=SVC(random_state=0))
51 classif.fit(X, y).predict(X)
52 print(classif.fit(X, y).predict(X))
53
54 y = LabelBinarizer().fit_transform(y)
55 classif.fit(X, y).predict(X)
56 print(classif.fit(X, y).predict(X))
57
58 y = y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
59 y = MultiLabelBinarizer().fit_transform(y)
60 print(classif.fit(X, y).predict(X))

```

## 1.3 Penanganan Error

Dari percobaan yang dilakukan di atas, apabila mendapatkan error maka:

1. skinsut error

```

ImportError: cannot import name 'load_iris' from
'sklearn' (E:\Software\Anaconda\lib\site-packages
\sklearn\__init__.py)

```

2. Tuliskan kode eror dan jenis errornya

(a) Import Error terjadi ketika suatu modul tidak ditemukan

3. Solusi pemecahan masalah error tersebut

(a) Pastikan memasukan nama modul yang tepat

```
import numpy as np
from sklearn import random_projection, svm, datasets
from sklearn.datasets import load_iris
from sklearn.multiclass import OneVsRestClassifier
from sklearn.preprocessing import LabelBinarizer
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.svm import SVC
```

# Chapter 2

## Membangun Model Prediksi

### 2.1 Teori

Praktek teori penunjang yang dikerjakan(nilai 5 per nomor, untuk hari pertama) :

1. Jelaskan apa itu binary classification dilengkapi ilustrasi gambar sendiri  
Binary classification adalah proses mengklasifikasi yang ouputnya dibagi menjadi 2 kelas

2. Jelaskan apa itu supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

*Supervised Learning* sebuah pembelajaran yang ditentukan berdasarkan penggunaan traning set yang berlabel.

*Unsupervised Learning* sebuah pembelajaran yang ditentukan berdasarkan penggunaan traning set yang tidak berlabel.

3. Jelaskan apa itu evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar sendiri

Evaluasi adalah tentang mengukur seberapa baik nilai performa dari suatu model. Dan akurasi adalah tigtat ketepatan yang benar dari suatu model.

4. Jelaskan bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix buatan sendiri.

Cara membuat dan membaca confusion matrix :

- (a) Tentukan studi kasus nya
- (b) Buat ke dalam Decision Tree
- (c) Siapkan data testing

- (d) Kemudian cari value dari variabel misal nya a,b,c,d
  - (e) Dan cari value dari recall, precision, accuracy dan juga error state
5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. Cara kerja dari K-fold cross validation :
- (a) Total instace dibagi menjadi n bagian
  - (b) Fold pertama merupakan bagian pertama yang menjadi Testing data dan sisanya menjadi training data
  - (c) Hitung akurasi nya dengan menggunakan persamaan
  - (d) Fold ke dua merupakan bagian kedua yang menjadi testing data dan sisanya menjadi training data
  - (e) Hitung lagi akurasi nya dan juga seterusnya pada fold yang terakhir
  - (f) Dan terakhir adalah hitung rata-rata akurasi nya

Training Data					Test Data
Fold1	Fold2	Fold3	Fold4	Fold5	
Fold1	Fold2	Fold3	Fold4	Fold5	
Fold1	Fold2	Fold3	Fold4	Fold5	
Fold1	Fold2	Fold3	Fold4	Fold5	
Fold2	Fold2	Fold3	Fold4	Fold5	

Figure 2.1: Ilutrasi K-fold cross validation

6. Jelaskan apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.  
Decision Tree adalah suatu metode yang digunakan untuk mengambil keputusan
7. Jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.  
information gain adalah kumpulan informasi yang diperoleh dari variable acak.  
Dan Entropi adalah tingkat keacakan pada informasi yang sedang diproses.

## 2.2 scikit-learn

Dataset ambil di <https://github.com/PacktPublishing/Python-Artificial-Intelligence-Projects-for-Beginners> folder Chapter01. Tugas anda adalah, dataset ganti menggunakan **student-mat.csv** dan mengganti semua nama variabel dari kode di bawah ini dengan nama-nama makanan (NPM mod 3=0), kota (NPM mod 3=1), buah (NPM

mod 3=2), . Jalankan satu per satu kode tersebut di spyder dengan menggunakan textitRun current cell. Kemudian Jelaskan dengan menggunakan bahasa yang mudah dimengerti dan bebas plagiat dan wajib skrinsut dari komputer sendiri masing masing nomor di bawah ini(nilai 5 masing masing pada hari kedua).

1. load dataset student-mat.csv

```
#1194014 : mod 3 = 2 (jadi nama buah)
import pandas as pd
import numpy as np
from sklearn import tree
import graphviz
from sklearn.model_selection import cross_val_score
import matplotlib.pyplot as plt
```

Figure 2.2: Import

```
##% no 1 Load dataset
mentimun = pd.read_csv('student-mat.csv', sep=';')
len(mentimun)
print(len(mentimun))
```

Figure 2.3: Source Code Task 1

```
In [2]: runcell('no 1 Load dataset', 'E:/kuliah/
pythonagain/Chapter2/Chapter2.py')
395

In [3]:
```

Figure 2.4: Hasil Task



## 2. generate binary label

```
## no 2 generate binary label (pass)
# (test grades, each 0-20 pts); threshold for passing is sum=30

mentimun['pass'] = mentimun.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
mentimun = mentimun.drop(['G1', 'G2', 'G3'], axis=1)
mentimun.head(5)
print(mentimun.head())
```

Figure 2.5: Source Code Task 2

```
In [4]: runcell('no 2 generate binary label (pass)', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
school sex age address famsize ... Dalc Walc health absences pass
0 GP F 18 U GT3 ... 1 1 3 6 0
1 GP F 17 U GT3 ... 1 1 3 4 0
2 GP F 15 U LE3 ... 2 3 3 10 0
3 GP F 15 U GT3 ... 1 1 5 2 1
4 GP F 16 U GT3 ... 1 2 5 4 0
[5 rows x 31 columns]
```

Figure 2.6: Hasil Task 2

## 3. use one-hot encoding on categorical columns

```
## no 3 use one-hot encoding on categorical columns
mentimun = pd.get_dummies(mentimun, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
                                             'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
                                             'nursery', 'higher', 'internet', 'romantic'])
mentimun.head(5)
print(mentimun.head())
```

Figure 2.7: Source Code Task 3

```
In [5]: runcell('no 3 use one-hot encoding on categorical columns', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
age Medu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 1 1 ... 1 1 0
2 15 1 1 ... 1 1 0
3 15 4 2 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]
```

Figure 2.8: Hasil Task 3

## 4. shuffle rows

```
mentimun = mentimun.sample(frac=1)
# split training and testing data
mentimun_train = mentimun[:500]
mentimun_test = mentimun[500:]

mentimun_train_att = mentimun_train.drop(['pass'], axis=1)
mentimun_train_pass = mentimun_train['pass']

mentimun_test_att = mentimun_test.drop(['pass'], axis=1)
mentimun_test_pass = mentimun_test['pass']

mentimun_att = mentimun.drop(['pass'], axis=1)
mentimun_pass = mentimun['pass']

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

Figure 2.9: Source Code Task 4

```
In [6]: runcell('no 4 shuffle rows', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
Passing: 166 out of 395 (42.03%)
```

Figure 2.10: Hasil Task 4

5. fit a decision tree

```
##% no 5 fit a decision tree
melon = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
melon = melon.fit(mentimun_train_att, mentimun_train_pass)
```

Figure 2.11: Source Code Task 5

```
In [13]: melon = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: melon = melon.fit(mentimun_train_att, mentimun_train_pass)
```

Figure 2.12: Hasil Task 5

6. visualize tree

```
##% no 6 visualize tree

dot_data = tree.export_graphviz(melon, out_file=None, label="all",
                                impurity=False, proportion=True,
                                feature_names=list(mentimun_train_att),
                                class_names=["fail", "pass"],
                                filled=True, rounded=True)

graph = graphviz.Source(dot_data)
print(graph)
```

Figure 2.13: Source Code Task 6

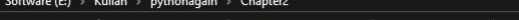
```
In [14]: runcell('no 6 visualize tree', 'E:/kuliah/pythonagain/Chapter2/
Chapter2.py')
digraph Tree {
    node [shape=box, style=filled, rounded, color="black", fontname="helvetica"] ;
    edge [fontname="helvetica"] ;
    0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\n\nclass = fail",
    fillcolor="#f8dccc"] ;
    1 [label="schoolsup_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\n\nclass =
    fail", fillcolor="#fdf6f0"] ;
    0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
    2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\n\nclass =
    fail", fillcolor="#eb9c63"] ;
    1 -> 2 ;
    3 [label="reason_reputation <= 0.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\n\nclass
    = fail", fillcolor="#e78c49"] ;
    2 -> 3 ;
    4 [label="samples = 4.3%\nvalue = [1.0, 0.0]\n\nclass = fail", fillcolor="#e58139"] ;
    3 -> 4 ;
    5 [label="Fedu <= 2.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\n\nclass = fail",
    fillcolor="#eca572"] ;
    4 -> 5 ;
    6 [label="samples = 1.0%\nvalue = [0.5, 0.5]\n\nclass = fail", fillcolor="ffffff"] ;
    5 -> 6 ;
    7 [label="samples = 1.3%\nvalue = [1.0, 0.0]\n\nclass = fail", fillcolor="#e58139"] ;
    6 -> 7 ;
    8 [label="health <= 2.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\n\nclass = fail",
    fillcolor="#f3c7a7"] ;
    7 -> 8 ;
}
```

Figure 2.14: Hasil Task 6

## 7. save tree

```
%% no 7 save tree
tree.export_graphviz(melon, out_file="student-performance.dot",
label="all", impurity=False,
proportion=True,
feature_names=list(mentimun_train_att),
class_names=["fail", "pass"],
filled=True, rounded=True)
```

Figure 2.15: Source Code Task 7



This PC > Software (E:) > Kuliah > pythonagain > Chapter2

Name	Date modified	Type	Size
.idea	3/24/2022 1:39 PM	File folder	
Chapter2	3/24/2022 7:44 PM	Python File	4 KB
student-mat	3/24/2022 1:18 PM	Microsoft Excel C...	57 KB
student-performance	3/24/2022 7:55 PM	Microsoft Word 9...	5 KB

Figure 2.16: Hasil Task 7

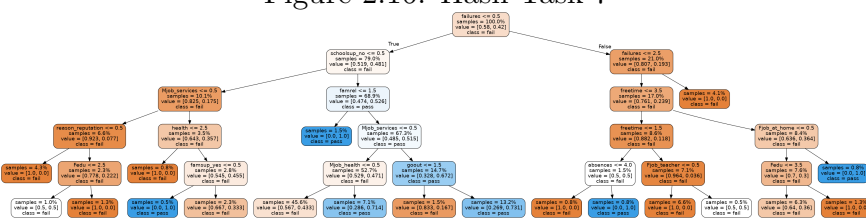


Figure 2.17: Hasil Task 7

8. task 8

```
#%% no 8
melon.score(mentimun_test_att, mentimun_test_pass)
print(melon.score(mentimun_test_att, mentimun_test_pass))
```

Figure 2.18: Source Code Task 8

```
In [16]: runcell('no 8', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
0.7063291139240506
```

Figure 2.19: Hasil Task 8

## 9. task 9

```
##%% no 9
scores = cross_val_score(melon, mentimun_att, mentimun_pass, cv=5)
# show average score and +/- two standard deviations away |
#(covering 95% of scores)
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

Figure 2.20: Source Code Task 9

```
In [17]: runcell('no 9', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
Accuracy: 0.59 (+/- 0.03)
```

Figure 2.21: Hasil Task 9

## 10. task 10

```

%% no 11

depth_acc = np.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    melon = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=max_depth)
    cau = cross_val_score(melon, mentimun_att, mentimun_pass, cv=5)
    depth_acc[i,0] = max_depth
    depth_acc[i,1] = cau.mean()
    depth_acc[i,2] = cau.std() * 2
    i += 1

print(depth_acc)

```

Figure 2.22: Source Code Task 10

```

In [10]: runcell('no 10', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.59 (+/- 0.05)
Max depth: 3, Accuracy: 0.59 (+/- 0.07)
Max depth: 4, Accuracy: 0.60 (+/- 0.04)
Max depth: 5, Accuracy: 0.59 (+/- 0.03)
Max depth: 6, Accuracy: 0.59 (+/- 0.08)
Max depth: 7, Accuracy: 0.56 (+/- 0.11)
Max depth: 8, Accuracy: 0.58 (+/- 0.09)
Max depth: 9, Accuracy: 0.55 (+/- 0.09)
Max depth: 10, Accuracy: 0.59 (+/- 0.09)
Max depth: 11, Accuracy: 0.59 (+/- 0.08)
Max depth: 12, Accuracy: 0.56 (+/- 0.09)
Max depth: 13, Accuracy: 0.56 (+/- 0.10)
Max depth: 14, Accuracy: 0.56 (+/- 0.07)
Max depth: 15, Accuracy: 0.57 (+/- 0.08)
Max depth: 16, Accuracy: 0.56 (+/- 0.09)
Max depth: 17, Accuracy: 0.57 (+/- 0.05)
Max depth: 18, Accuracy: 0.59 (+/- 0.05)
Max depth: 19, Accuracy: 0.58 (+/- 0.10)

```

Figure 2.23: Hasil Task 10

## 11. task 11

```

%% no 11

depth_acc = np.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    melon = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=max_depth)
    cau = cross_val_score(melon, mentimun_att, mentimun_pass, cv=5)
    depth_acc[i,0] = max_depth
    depth_acc[i,1] = cau.mean()
    depth_acc[i,2] = cau.std() * 2
    i += 1

print(depth_acc)

```

Figure 2.24: Source Code Task 11

```

In [19]: runcell('no 11', 'E:/Kuliah/pythonagain/Chapter2/Chapter2.py')
[[1.00000000e+00 5.79746835e-01 1.01265823e-02]
 [2.00000000e+00 5.89873418e-01 5.45333145e-02]
 [3.00000000e+00 5.89873418e-01 6.71721477e-02]
 [4.00000000e+00 6.05063291e-01 4.05063291e-02]
 [5.00000000e+00 5.87341772e-01 5.21297729e-02]
 [6.00000000e+00 5.97468354e-01 8.06955820e-02]
 [7.00000000e+00 5.62025316e-01 1.18312116e-01]
 [8.00000000e+00 5.77215190e-01 1.08122311e-01]
 [9.00000000e+00 5.67088608e-01 1.07884941e-01]
 [1.00000000e+01 5.94936709e-01 9.73943497e-02]
 [1.10000000e+01 5.59493671e-01 7.05234849e-02]
 [1.20000000e+01 5.59493671e-01 9.25350222e-02]
 [1.30000000e+01 5.59493671e-01 4.90904289e-02]
 [1.40000000e+01 5.59493671e-01 7.23182626e-02]
 [1.50000000e+01 5.51898734e-01 8.25797794e-02]
 [1.60000000e+01 5.77215190e-01 7.26718992e-02]
 [1.70000000e+01 5.77215190e-01 9.28116596e-02]
 [1.80000000e+01 5.62025316e-01 8.56280229e-02]
 [1.90000000e+01 5.87341772e-01 1.06930188e-01]]

```

Figure 2.25: Hasil Task 11

## 12. task 12

```
%% 12
fig, ax = plt.subplots()
ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
plt.show()
```

Figure 2.26: Source Code Task 12

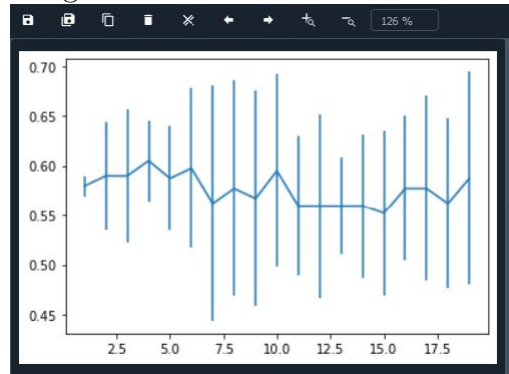


Figure 2.27: Hasil Task 12

## 2.3 Penanganan Error

Dari percobaan yang dilakukan di atas, error yang kita dapatkan di dokumentasikan dan di selesaikan (nilai 5 hari kedua):

1. skrinsut error
2. Tuliskan kode eror dan jenis errornya
3. Solusi pemecahan masalah error tersebut

## Chapter 3

# Prediksi dengan Random Forest

Untuk pratikum saati ini menggunakan buku *Python Artificial Intelligence Projects for Beginners*[?]. Dengan praktek menggunakan python 3 dan editor anaconda dan library python scikit-learn. Kode program ada di <https://github.com/PacktPublishing/Python-Artificial-Intelligence-Projects-for-Beginners> . Tujuan pembelajaran pada pertemuan pertama antara lain:

1. Mengerti implementasi klasifikasi dan teknik evaluasi
2. Memprediksi spesies burung dengan random forest
3. Memahami Confusion Matrix.

Tugas dengan cara dikumpulkan dengan pull request ke github dengan menggunakan latex pada repo yang dibuat oleh asisten riset. Kode program menggunakan input listing ditaruh di folder src ekstensi .py dan dipanggil ke latex dengan input listings. Tulisan dan kode tidak boleh plagiat, menggunakan bahasa indonesia yang sesuai dengan gaya bahasa buku teks.

### 3.1 Teori

Random Forest adalah hasil voting dari beberapa decision tree yang masing-masing memegang atribut yang berbeda. Jadi setiap decision tree spesifik terhadap atribut tersebut yang merupakan bagian kecil dari keseluruhan atribut di data set. Hindari RF jika atribut terlalu sedikit untuk membentuk beberapa tree. Pada praktek kali ini menggunakan dataset spesies burung yang diambil dari situs (<http://www.vision.caltech.edu/visip200-2011.html>). Didalamnya terdapat 12.000 foto dari 200 spesies yang berbeda. Yang akan kita pakai untuk RF hanya atribut dari burungnya saja seperti ukuran,

bentuk dan warna. Data tersebut diberi label secara manual oleh manusia dengan memanfaatkan jasa dari Amazon's Mechanical Turk.

### 3.1.1 Random Forest

Pertama dataset kita baca terlebih dahulu.

```
1 import pandas as pd
2
3 # some lines have too many fields (?), so skip bad lines
4 imgatt = pd.read_csv("data/CUB_200_2011/attributes/
    image_attribute_labels.txt",
5                      sep='\s+', header=None, error_bad_lines=False,
6                      warn_bad_lines=False,
7                      usecols=[0,1,2], names=['imgid', 'attid', 'present'
8                      ])
9
```

Listing 3.1: Membaca data file txt

Melihat sebagian data awal, dengan menggunakan listing 3.2.

```
1 imgatt.head()
```

Listing 3.2: Melihat sebagian data awal

Melihat jumlah data menggunakan listing 3.3.

```
1 imgatt.shape
```

Listing 3.3: Mengetahui jumlah data

Merubah atribut menjadi kolom dengan menggunakan pivot layaknya excel. lalu kita cek isinya dengan menggunakan perintah pada listing 3.4.

```
1 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
2
3 imgatt2.head()
4 imgatt2.shape
```

Listing 3.4: Pivot dataset

Sekarang kita akan meload jawabannya yang berisi apakah burung itu termasuk dalam spesies yang mana. Dua kolomnya adalah imgid dan label. Dan melakukan pivot yang mana imgid menjadi index yang artinya unik perintahnya ada di listing 3.5. Lalu kita cek kembali datanya.

```
1 imglabels = pd.read_csv("data/CUB_200_2011/image_class_labels.txt",
2                          sep=' ', header=None, names=['imgid', 'label'])
3
4 imglabels = imglabels.set_index('imgid')
5
6
7 imglabels.head()
8 imglabels.shape
```

Listing 3.5: membaca dataset label file txt

Karena isinya sama kita bisa melakukan join antara dua data. Sehingga kita akan mendapatkan data ciri dan data jawabannya atau labelnya sehingga bisa dikategorikan supervised learning. maka perintah untuk menggabungkan kedua data dan kemudian kita melakukan pemisahan antara data set untuk training dan test dengan perintah di listing 3.6.

```
1 df = imgatt2.join(imglabels)
2 df = df.sample(frac=1)
```

Listing 3.6: Menggabungkan field dari dua file terpisah

Kemudian drop label yang didepan, dan gunakan label yang paling belakang yang baru di join dengan perintah listing 3.7.

```
1 df_att = df.iloc[:, :312]
2 df_label = df.iloc[:, 312:]
```

Listing 3.7: Memisahkan dan memilih label

Kita bisa mengecek isinya dengan perintah listing 3.8.

```
1 df_att.head()
2 df_label.head()
```

Listing 3.8: Melihat isi masing masing data frame

Kita bagi menjadi dua bagian, 8000 row pertama sebagai data training sisanya sebagai data testing dengan perintah listing 3.9.

```
1 df_train_att = df_att[:8000]
2 df_train_label = df_label[:8000]
3 df_test_att = df_att[8000:]
4 df_test_label = df_label[8000:]
5
6 df_train_label = df_train_label['label']
7 df_test_label = df_test_label['label']
```

Listing 3.9: Pembagian data training dan test

Kita panggil kelas RandomForestClassifier. max features diartikan sebagai berapa banyak kolom pada setiap tree dengan perintah listing 3.10.

```
1 from sklearn.ensemble import RandomForestClassifier
2 clf = RandomForestClassifier(max_features=50, random_state=0,
    n_estimators=100)
```

Listing 3.10: Instansiasi kelas Random Forest

Kemudian lakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50 untuk perpohonnya dengan perintah listing 3.11.

```
1 clf.fit(df_train_att, df_train_label)
```

Listing 3.11: Fitting random forest dengan dataset training



Hasilnya bisa kita dapatkan dengan perintah predict dengan perintah listing 3.12.

```
1 print(clf.predict(df_train_att.head()))
```

Listing 3.12: Melihat Hasil prediksi

Untuk besaran akurasi dengan perintah listing 3.13

```
1 clf.score(df_test_att, df_test_label)
```

Listing 3.13: Score perolehan dari klasifikasi

### 3.1.2 Confusion Matrix

Dari RF kita coba petakan ke dalam Confusion Matrix dan lihat hasilnya dengan perintah listing 3.14.

```
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
5 cm
```

Listing 3.14: Membuat Confusion Matrix

Kemudian kita plot dengan perintah

```
1 import matplotlib.pyplot as plt
2 import itertools
3 def plot_confusion_matrix(cm, classes,
4                           normalize=False,
5                           title='Confusion matrix',
6                           cmap=plt.cm.Blues):
7     """
8     This function prints and plots the confusion matrix.
9     Normalization can be applied by setting 'normalize=True'.
10    """
11    if normalize:
12        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13        print("Normalized confusion matrix")
14    else:
15        print('Confusion matrix, without normalization')
16
17    print(cm)
18
19    plt.imshow(cm, interpolation='nearest', cmap=cmap)
20    plt.title(title)
21    #plt.colorbar()
22    tick_marks = np.arange(len(classes))
23    plt.xticks(tick_marks, classes, rotation=90)
24    plt.yticks(tick_marks, classes)
25
26    fmt = '.2f' if normalize else 'd'
27    thresh = cm.max() / 2.
```

```

28     #for i, j in itertools.product(range(cm.shape[0]), range(cm.shape
    [1])):
29     #     plt.text(j, i, format(cm[i, j], fmt),
30     #               horizontalalignment="center",
31     #               color="white" if cm[i, j] > thresh else "black")
32
33     plt.tight_layout()
34     plt.ylabel('True label')
35     plt.xlabel('Predicted label')

```

Listing 3.15: Plotting Confusion Matrix

Agar plot sumbunya sesuai dengan nama datanya maka kita set dengan perintah

```

1 birds = pd.read_csv("data/CUB_200_2011/classes.txt",
2                     sep='\s+', header=None, usecols=[1], names=['
    birdname'])
3 birds = birds['birdname']
4 birds

```

Listing 3.16: Membaca file classes.txt

Lalu kita plot

```

1 import numpy as np
2 np.set_printoptions(precision=2)
3 plt.figure(figsize=(60,60), dpi=300)
4 plot_confusion_matrix(cm, classes=birds, normalize=True)
5 plt.show()

```

Listing 3.17: Plot hasil perubahan label

### 3.1.3 Mencoba dengan metode Decission Tree dan SVM

Kita coba menggunakan Decission tree

```

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)

```

Listing 3.18: Mencoba klasifikasi dengan decission tree dengan dataset yang sama

Kita coba menggunakan SVM

```

1 from sklearn import svm
2 clfsvm = svm.SVC()
3 clfsvm.fit(df_train_att, df_train_label)
4 clfsvm.score(df_test_att, df_test_label)

```

Listing 3.19: Mencoba klasifikasi dengan SVM dengan dataset yang sama

### 3.1.4 Pengecekan Cross Validation

Pengecekan Cross Validation untuk random forest

```
1 from sklearn.model_selection import cross_val_score
2 scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
3 # show average score and +/- two standard deviations away (covering 95%
  of scores)
4 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

Listing 3.20: Hasil Cross Validation random forest

untuk decision tree

```
1 scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv
  =5)
2 print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std
  () * 2))
```

Listing 3.21: Hasil Cross Validation Decision Tree

untuk SVM

```
1 scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
2 print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std()
  * 2))
```

Listing 3.22: Hasil Cross Validation SVM

### 3.1.5 Pengamatan komponen informasi

Untuk mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya menggunakan kode

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

Listing 3.23: Melakukan Pengamatan komponen informasi

Dan kita bisa melakukan plot informasi ini dengan kode

```
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 estimators')
14 ax.set_zlabel('Avg accuracy')
15 plt.show()
```

Listing 3.24: Plot Komponen informasi agar bisa dibaca

## 3.2 Soal Teori

Praktek teori penunjang yang dikerjakan(nilai 5 per nomor, untuk hari pertama) :

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.
2. Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.
3. Jelaskan apa itu cross validation
4. Jelaskan apa arti score 44% pada random forest, 27% pada decision tree dan 29%dari SVM.
5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.
6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

## 3.3 Praktek Program

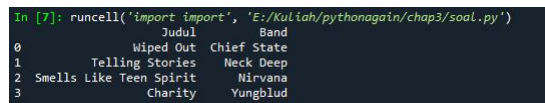
Tugas anda adalah,praktekkan dan jelaskan dengan menggunakan bahasa yang mudah dimengerti dan bebas plagiat dan wajib skrinsut dari komputer sendiri masing masing nomor di bawah ini(nilai 5 masing masing pada hari kedua).

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

```

1  """ 1. Pandas sederhana = 2 dimensional struktur data
2  lagu = {'Judul': ['Wiped Out', 'Telling Stories', 'Smells Like
Teen Spirit', 'Charity'],
3         'Band': ['Chief State', 'Neck Deep', 'Nirvana', '
Yungblud']} # dictionary yang menampung key dan value nya berupa
judul dan band
4
5  df = pd.DataFrame(lagu) #kemudian dibuat ke dalam bentuk
dataframe yang tampung kedalam sebuah variable dengan nama df
6
7  print(df) #untuk menampilkan menggunakan print yang diarahkan
ke variable df tadi

```



```

In [7]: runcell('import import', 'E:/Kuliah/pythonagain/chap3/soal.py')

```

	Judul	Band
0	Wiped Out	Chief State
1	Telling Stories	Neck Deep
2	Smells Like Teen Spirit	Nirvana
3	Charity	Yungblud

Figure 3.1: Hasil Pandas Sederhana

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

```

1  """ 2. Numpy sederhana
2  fa = np.array([14,12,20]) # variable fa untuk menampung array 1
dimensi
3  print(fa) #untuk menampilkan hasil variable fa

```



```

In [10]: runcell('2. Numpy sederhana',
'E:/Kuliah/pythonagain/chap3/soal.py')
[14 12 20]
In [11]:

```

Figure 3.2: Hasil Numpy Sederhana

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

```

1  """ 3. Matplotlib sederhana = untuk memvisualisasikan data ke
dalam bentuk grafik
2
3  x = [1,2,3,5,6,7,3] #variable x untuk menampung semua value dan
juga sebagai sumbu x
4  y = [1,2,3,3,2,4,5] #variable y untuk menampung semua value dan
juga sebagai sumbu y
5
6  plt.plot(x, y) #plotting variable x dan y
7
8  plt.xlabel('sumbu x') #memberikan label untuk variable x

```

```

9     plt.ylabel('sumbu y') #memberikan label untuk variabel y
10
11     plt.title('Matplotlib sederhana') #memberi title/judul pada
    plotting yang dibuat
12
13     plt.show() #method show digunakan untuk menampilkan plot

```

```

In [10]: runcell('2. Numpy sederhana',
'E:/Kuliah/pythonagain/chap3/soal.py')
[14 12 20]
In [11]:

```

Figure 3.3: Hasil Numpy Sederhana

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

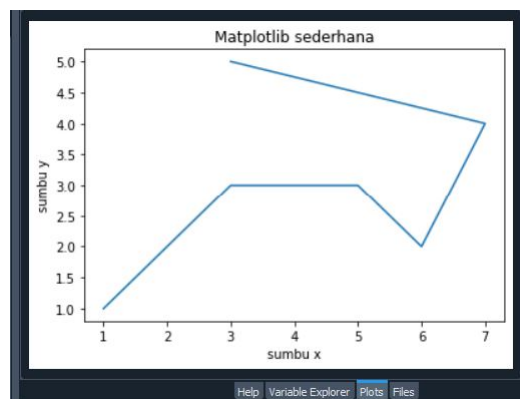


Figure 3.4: Hasil Random Forest

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

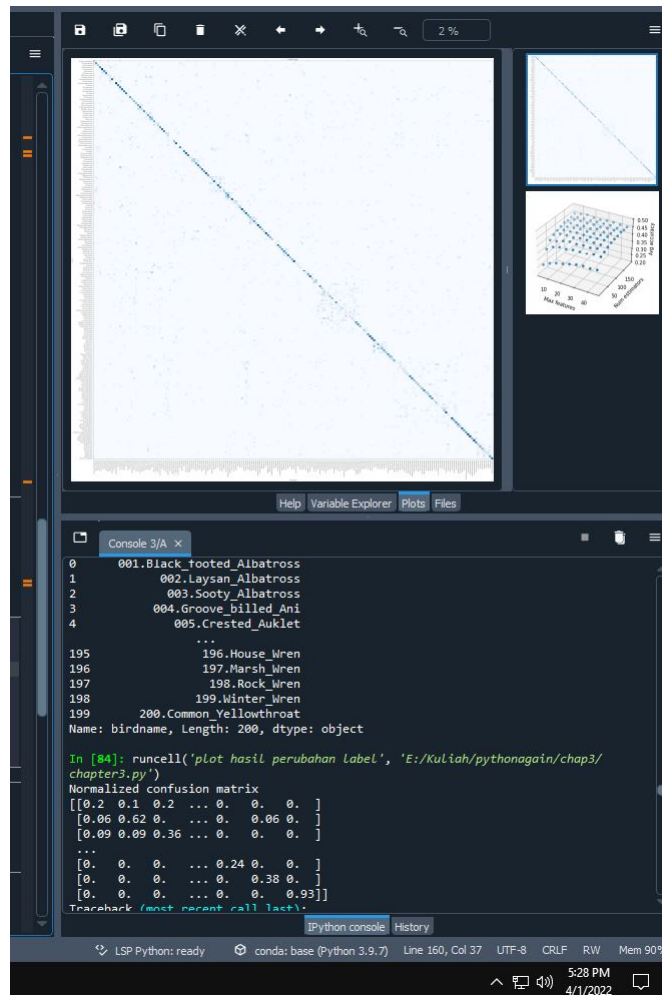
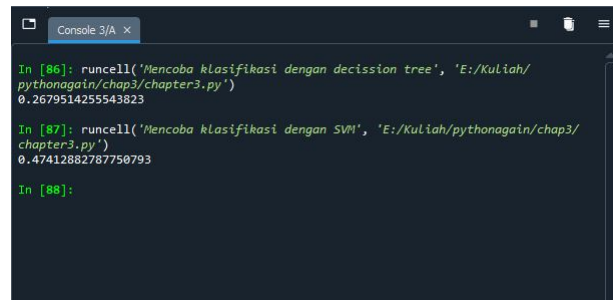


Figure 3.5: Hasil ConfusionMatrix

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.



```
Console 3/A x

In [86]: runcell('Mencoba klasifikasi dengan decission tree', 'E:/Kuliah/
pythonagain/chap3/chapter3.py')
0.2679514255543823

In [87]: runcell('Mencoba klasifikasi dengan SVM', 'E:/Kuliah/pythonagain/chap3/
chapter3.py')
0.47412882787750793

In [88]:
```

Figure 3.6: Hasil Klasifikasi DT

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



```

0.40000000000000002
In [2]: runcell('Hasil cross validation
untuk random forest', 'E:/Kuliah/
pythonagain/chap3/chapter3.py')
Accuracy: 0.44 (+/- 0.04)

In [3]: |

```

Figure 3.7: Hasil Cross Validation Random Forest

```

In [3]: runcell('Hasil cross validation
untuk decision tree', 'E:/Kuliah/
pythonagain/chap3/chapter3.py')
Accuracy: 0.26 (+/- 0.03)

In [4]:

```

Figure 3.8: Hasil Cross Validation Klasifikasi DT

```

In [4]: runcell('Hasil cross validation
untuk SVM', 'E:/Kuliah/pythonagain/
chap3/chapter3.py')
Accuracy: 0.47 (+/- 0.02)

In [5]:

```

Figure 3.9: Hasil Cross Validation SVM

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

```

In [62]: runcell('melakukan pengamatan komponen informasi', 'E:/Kuliah/
pythonagain/chap3/chapter3.py')
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.02)
Max features: 5, num estimators: 50, accuracy: 0.40 (+/- 0.01)
Max features: 5, num estimators: 70, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.43 (+/- 0.01)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.01)
Max features: 5, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.02)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.02)
Max features: 10, num estimators: 50, accuracy: 0.40 (+/- 0.01)
Max features: 10, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 90, accuracy: 0.44 (+/- 0.01)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.01)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)

```

Figure 3.10: Hasil pengamatan komponen informasi

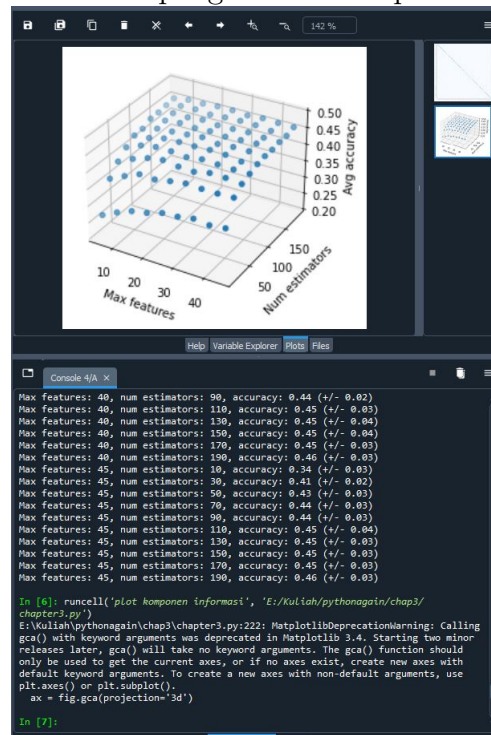


Figure 3.11: Hasil Plot Komponen Informasi

## 3.4 Penanganan Error

Dari percobaan yang dilakukan di atas, error yang kita dapatkan di dokumentasikan dan di selesaikan(nilai 5 per error yang ditangani. Untuk hari kedua):

1. skripsi error

2. Tuliskan kode eror dan jenis errornya
3. Solusi pemecahan masalah error tersebut

### 3.5 Presentasi Tugas

Pada pertemuan ketiga ini, diadakan tiga penilaiain yaitu penilaian untuk tugas mingguan seperti sebelumnya dengan nilai maksimal 100. Kemudian dalam satu minggu kedepan maksimal sebelum waktu mata kuliah kecerdasan buatan. Ada presentasi tugas bab ini dan bab sebelumnya dengan nilai presentasi yang terpisah masing-masing 100. Jadi ada tiga komponen penilaiain pada pertemuan ini yaitu :

1. tugas minggu hari ini dan besok (maks 100). pada chapter ini
2. presentasi decission tree (maks 100). Mempraktekkan kode python dan menjelaskan cara kerjanya.
3. presentasi Random Forest (maks 100).Mempraktekkan kode python dan menjelaskan cara kerjanya.

Waktu presentasi pada jam kerja di IRC. Kriteria penilaian presentasi sangat sederhana, jika presenter tidak bisa menjawab pertanyaan asisten maka nilai nol. Jika semua pertanyaan bisa dijawab maka nilai 100. Presentasi bisa diulang apabila nilai nol sampai bisa mendapatkan nilai 100 dalam waktu satu minggu kedepan.