

# **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 Tunnisa

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. Sariasih 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 izin 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>57</b>
<b>3 Chapter 3</b>	<b>71</b>
<b>4 Chapter 4</b>	<b>73</b>
<b>5 Chapter 5</b>	<b>75</b>
<b>6 Chapter 6</b>	<b>77</b>
<b>7 Chapter 7</b>	<b>79</b>





# DAFTAR ISI

---

Daftar Gambar	xiii
Daftar Tabel	xvii
Foreword	xxiii
Kata Pengantar	xxv
Acknowledgments	xxvii
Acronyms	xxix
Glossary	xxxi
List of Symbols	xxxiii
Introduction	xxxv
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
<b>1 Chapter 1</b>	<b>1</b>
1.1 1174035 - Luthfi Muhammad Nabil	1
1.1.1 Sejarah dan perkembangan Kecerdasan Buatan	1
1.1.2 Supervised Learning	2
1.1.3 Unsupervised Learning	2
	<b>ix</b>

1.1.4	Jenis - Jenis Dataset	3
1.1.5	Instalasi dan Percobaan Kompilasi dari Library Scikit-learn	3
1.1.6	Mencoba Loading and example dataset	5
1.1.7	Learning and Predicting	7
1.1.8	Model Persistence	8
1.1.9	Conventions	10
1.1.10	Skrinsut Error	14
1.1.11	Kode error dan jenis error tersebut	14
1.1.12	Penanganan Error	15
1.1.13	Plagiarisme	16
1.2	1174040 - Hagan Rowlenstino A. S	16
1.2.1	Teori	16
1.2.2	Instalasi	17
1.2.3	Penanganan Error	20
1.2.4	Cek Plagiarism	21
1.3	1174042 Faisal Najib Abdullah	21
1.3.1	Teori	21
1.3.2	Instalasi	22
1.3.3	Penanganan eror	27
1.3.4	Plagiat	28
1.3.5	Link	28
1.4	1174043 - Irvan Rizkiansyah	29
1.4.1	Definisi Kecerdasan Buatan	29
1.4.2	Sejarah dan Perkembangan	29
1.4.3	Supervised Learning	29
1.4.4	Unsupervised Learning	30
1.4.5	Teknik Klasifikasi	30
1.4.6	Regresi	30
1.4.7	Training Set	30
1.4.8	Testing Set	30
1.4.9	Instalasi dan Percobaan Kompilasi dari Library Scikit-learn	30
1.4.10	Mencoba Loading an example dataset	32
1.4.11	Mencoba Learning and Predicting	33
1.4.12	Mencoba Model Persistence	33
1.4.13	Mencoba Conventions	34
1.5	1174050 Dika Sukma Pradana	34

1.5.1	Teori	34
1.5.2	Instalasi	36
1.5.3	Percobaan	36
1.5.4	Penanganan eror	41
1.5.5	Plagiarism	42
1.6	1174057 Alit Fajar Kurniawan	42
1.6.1	Teori	42
1.6.2	Praktek	43
1.6.3	Penanganan Error	49
1.6.4	Bukti Tidak Plagiat	50
1.7	1174039 - Liyana Majdah Rahma	51
1.7.1	Teori	51
1.7.2	Instalasi	52
1.7.3	Penanganan Error	55
1.7.4	Cek Plagiarism	56
<b>2</b>	<b>Chapter 2</b>	<b>57</b>
2.1	1174042 Faisal Najib Abdullah	57
2.1.1	Teori	57
2.1.2	Sikic-Learn	62
2.1.3	Penanganan Error	68
<b>3</b>	<b>Chapter 3</b>	<b>71</b>
<b>4</b>	<b>Chapter 4</b>	<b>73</b>
<b>5</b>	<b>Chapter 5</b>	<b>75</b>
<b>6</b>	<b>Chapter 6</b>	<b>77</b>
<b>7</b>	<b>Chapter 7</b>	<b>79</b>



# DAFTAR GAMBAR

---

1.1	Instalasi Scikit Learn	3
1.2	Daftar Example	3
1.3	Variable Explorer	4
1.4	Hasil Percobaan 1	5
1.5	Hasil Percobaan 2	5
1.6	Hasil Percobaan 3	6
1.7	Hasil Percobaan 4	6
1.8	Hasil pada variable explorer	6
1.9	Hasil Percobaan 1	7
1.10	Hasil Percobaan 2	7
1.11	Hasil Percobaan 3	7
1.12	Hasil pada variable explorer	8
1.13	Hasil Percobaan 1	8

1.14	Hasil Percobaan 2	9
1.15	Hasil Percobaan 3	9
1.16	Hasil Percobaan 4	9
1.17	Hasil Percobaan 5	9
1.18	Hasil pada variable explorer	10
1.19	Hasil Percobaan 1	11
1.20	Hasil Percobaan 2	11
1.21	Hasil Percobaan 3	11
1.22	Hasil Percobaan 4	12
1.23	Hasil Percobaan 5	12
1.24	Hasil Percobaan 6	12
1.25	Hasil pada variable explorer	13
1.26	Hasil Percobaan 6	14
1.27	Hasil pada variable explorer	16
1.28	Install Library Scikit	17
1.29	Variable Explorer	18
1.30	Data Digits	18
1.31	Digits Target	19
1.32	Data 2D	19
1.33	Data 2D	20
1.34	Cek Plagiarism	21
1.35	Installasi	23
1.36	Mencoba Loading an example Dataset	24
1.37	Learning and Predicting	24
1.38	Model Presistence	25
1.39	Model Presistence	25
1.40	Model Presistence	26
1.41	Error	28

1.42	Error	28
1.43	Instalasi Scikit Learn	31
1.44	Variable Explorer	32
1.45	Dataset	33
1.46	Predicting	33
1.47	Instalasi	36
1.48	Variabel Explore	37
1.49	Datasets	37
1.50	Error	41
1.51	Plagiarism	42
1.52	Instalasi Scikit Learn	44
1.53	Example	44
1.54	Example	45
1.55	Result Data Digits	46
1.56	Result digits.target	46
1.57	Result digits.image	46
1.58	Result Learning and predicting	47
1.59	Result Model persistence	48
1.60	Result Conventions	49
1.61	Error	49
1.62	Error	50
1.63	Plagiarisme	50
1.64	Instalasi	52
1.65	Variable Exploler	53
1.66	Data Digits	53
1.67	Digits Target	53
1.68	Data 2D	53
1.69	Data 2D	55



1.70	Cek Plagiarism	56
2.1	contoh binari calssification	58
2.2	contoh supervised learning	58
2.3	contoh unsupervised learning	59
2.4	contoh clusterring	59
2.5	contoh evaluasi dan akurasi	60
2.6	contoh Confusion Matrix	60
2.7	contoh K-fold cross validation	61
2.8	contoh decision tree	62
2.9	contoh information gain	62
2.10	hasil	63
2.11	hasil	63
2.12	hasil	64
2.13	hasil	65
2.14	hasil	65
2.15	hasil	66
2.16	hasil	66
2.17	hasil	67
2.18	hasil	67
2.19	hasil	68
2.20	hasil	69
2.21	hasil	69

# DAFTAR TABEL

---



# Listings

---

src/1174035/chapter1/sample1.py	4
src/1174035/chapter1/sample2.py	5
src/1174035/chapter1/sample2.py	5
src/1174035/chapter1/sample2.py	5
src/1174035/chapter1/sample2.py	6
src/1174035/chapter1/sample2.py	6
src/1174035/chapter1/sample3.py	7
src/1174035/chapter1/sample3.py	7
src/1174035/chapter1/sample3.py	7
src/1174035/chapter1/sample3.py	8
src/1174035/chapter1/sample4.py	8
src/1174035/chapter1/sample4.py	9
src/1174035/chapter1/sample4.py	9
src/1174035/chapter1/sample4.py	9
src/1174035/chapter1/sample4.py	9
src/1174035/chapter1/sample4.py	10
src/1174035/chapter1/sample5.py	10
src/1174035/chapter1/sample5.py	11

src/1174035/chapter1/sample5.py	11
src/1174035/chapter1/sample5.py	12
src/1174035/chapter1/sample5.py	12
src/1174035/chapter1/sample5.py	12
src/1174035/chapter1/sample5.py	12
src/1174035/chapter1/sample5.py	14
src/1174035/chapter1/sample3.py	15
src/1174040/chap1/ex1.py	17
src/1174040/chap1/ex2.py	18
src/1174040/chap1/ex2.py	18
src/1174040/chap1/ex2.py	18
src/1174040/chap1/ex2.py	19
src/1174040/chap1/no3.py	19
src/1174040/chap1/no4.py	20
src/1174040/chap1/no5.py	20
src/1174040/chap1/no3.py	21
src/1174042/chapter1/2,1.py	22
src/1174042/chapter1/2,2.py	22
src/1174042/chapter1/2,3.py	23
src/1174042/chapter1/2,4.py	24
src/1174042/chapter1/2,5.py	25
src/1174043/chapter1/sample1.py	31
src/1174043/chapter1/sample2.py	32
src/1174043/chapter1/sample3.py	33
src/1174043/chapter1/sample4.py	33
src/1174043/chapter1/sample5.py	34
src/1174050/chapter1/VAR.py	36
src/1174050/chapter1/dataset.py	37
src/1174050/chapter1/learning.py	37
src/1174050/chapter1/modelpersistance.py	38
src/1174050/chapter1/typecasting.py	39
src/1174050/chapter1/Multiclass.py	40
src/1174050/chapter1/Refitting.py	40
src/1174057/chapter1/example.py	44
src/1174057/chapter1/dataset.py	45
src/1174057/chapter1/learning.py	47
src/1174057/chapter1/modelpersistence.py	47
src/1174057/chapter1/modelpersistence.py	48

src/1174039/chapter1/no1.py	52
src/1174039/chapter1/no2.py	53
src/1174039/chapter1/no2.py	53
src/1174039/chapter1/no2.py	53
src/1174039/chapter1/no2.py	53
src/1174039/chapter1/no3.py	54
src/1174039/chapter1/no4.py	54
src/1174039/chapter1/no5.py	55
src/1174039/chapter1/no3.py	55
src/1174042/chapter2/2,1.py	62
src/1174042/chapter2/2,2.py	63
src/1174042/chapter2/2,3.py	64
src/1174042/chapter2/2,4.py	64
src/1174042/chapter2/2,5.py	65
src/1174042/chapter2/2,6.py	65
src/1174042/chapter2/2,7.py	65
src/1174042/chapter2/2,8.py	66
src/1174042/chapter2/2,9.py	66
src/1174042/chapter2/2,10.py	67
src/1174042/chapter2/2,11.py	67
src/1174042/chapter2/2,12.py	68



# 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 Inter-ship.

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
  
- $\mathcal{B}$  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[?].

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



# BAB 1

---

## CHAPTER 1

---

### 1.1 1174035 - Luthfi Muhammad Nabil

Kecerdasan buatan merupakan kecerdasan yang dimasukkan ke sistem yang dapat diatur untuk kepentingan ilmiah. Kecerdasan buatan biasa disebut AI (Artificial Intelligence) yang didefinisikan sebagai kecerdasan ilmiah. AI memiliki kemampuan untuk menerjemahkan data dari luar, dan mempelajari data tersebut untuk dipelajari demi mencapai tujuan dan melakukan tugas tertentu sesuai hasil adaptasi berdasarkan data yang didapat.

#### 1.1.1 Sejarah dan perkembangan Kecerdasan Buatan

AI mulai berkembang sesuai dengan konsep yang dikemukakan pada awal abad 17, Rene Descartes menyebutkan bahwa tubuh hewan bukanlah apa-apa melainkan mesin-mesin yang rumit. Lalu Blaise Pascal menciptakan mesin perhitungan digital mekanis pertama pada 1642. Selanjutnya pada abad ke 19, Charles Babbage dan Ada Lovelace menciptakan sebuah mesin penghitung mekanis yang dapat diprogram.

Pada tahun 1950-an, Program AI pertama yang sudah dapat difungsikan telah ditulis pada 1951 untuk menjalankan mesin Ferranti Mark I di University of Manchester yang merupakan sebuah program permainan naskah yang ditulis oleh Christopher Strachey. John McCarthy menyebutkan istilah kecerdasan buatan pada konferensi pertama yang disediakan untuk persoalan ini. Dilanjut pada tahun 1956, Beliau menemukan bahasa pemrograman yang bernama Lisp.

Jaringan saraf mulai digunakan secara luas pada tahun 1980-an, dimana algoritma perambatan balik pertama kali dijelaskan oleh Paul John Werbos pada tahun 1974. Selanjutnya di tahun 1982, para ahli fisika menganalisis sifat dari penyimpanan dan optimasi pada jaringan saraf menggunakan sistem statistika. Lalu dilanjutkan pada tahun 1985 sedikitnya empat kelompok riset menemukan algoritma pembelajaran propagansi balik. Algoritma ini berhasil diimplementasikan ke ilmu komputer dan psikologi. Dan pada tahun 1990, ditandai perolehan besar dalam berbagai bidang AI dan demonstrasi dari berbagai aplikasi yang sudah mengimplementasi. Seperti Deep Blue, sebuah komputer dari permainan catur yang dapat mengalahkan Garry Kasparov dalam sebuah pertandingan 6 game yang terkenal pada 1997.

### 1.1.2 Supervised Learning

Supervised learning adalah kondisi yang menggunakan variabel input dan output untuk dapat dilakukan pemetaan input output yang sudah didapat. Disebut Supervised Learning karena proses dari pembelajaran algoritma dari pembelajaran yang disumbangkan dengan dataset dapat dipikirkan seperti seorang guru yang mengawasi proses pembelajaran. Proses pembelajaran dari algoritma akan berhenti saat algoritma sudah mendapatkan level dari performansi yang dapat diterima.

Masalah dari Supervised learning dapat dikelompokkan menjadi masalah dengan regresi dan klasifikasi

- Klasifikasi : Masalah dalam klasifikasi yang dimana output dari variable itu adalah kategori, seperti "Laki - laki" atau "Perempuan, dan "Muda" dan "Tua"
- Regresi : Masalah dalam regresi adalah jika pengeluaran dari variabel adalah sebuah nilai asli, seperti "suhu", dan "tinggi"

### 1.1.3 Unsupervised Learning

Unsupervised learning adalah kondisi dimana kamu hanya memiliki input data tanpa memiliki variabel output yang sesuai. Tujuan dari unsupervised learning adalah untuk memodelkan distribusi pada data untuk mengetahui lebih lanjut mengenai data. Disebut unsupervised learning karena pada metode ini, tidak ada jawaban yang tepat dan tidak ada pengaruh. Sehingga algoritma

akan ditinggalkan sesuai rancangan demi menemukan dan dapat mengolah data yang menarik pada saat yang akan datang.

### 1.1.4 Jenis - Jenis Dataset

Dataset merupakan objek yang merepresentasikan data dan relasinya di memor. Strukturnya dapat mirip sesuai dengan struktur yang ada pada database namun bisa diubah sesuai dengan kebutuhan. Dataset juga berisi koleksi dari tabel data dan relasi data.

- Training set : merupakan sebuah dataset yang digunakan untuk kepentingan pembelajaran. Kepentingan tersebut akan disesuaikan dengan parameter yang ada.
- Test dataset : adalah sebuah dataset yang bersifat independen dibandingkan dengan training dataset, namun mengikuti probabilitas distribusi yang sama dengan training dataset. Jika model sudah sesuai dengan training dataset maka dataset sudah dapat disesuaikan dengan test dataset. Penyesuaian dari training dataset .

### 1.1.5 Instalasi dan Percobaan Kompilasi dari Library Scikit-learn

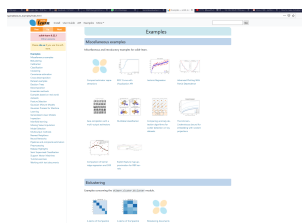
1. Buka anaconda prompt
2. Ketik di anaconda prompt yaitu : "pip install -U scikit-learn" untuk instalasi

```

(base) D:\VallanPython>cmd
(base) D:\VallanPython>pip install -U scikit-learn
Collecting scikit-learn
  Downloading scikit-learn-0.22.1-py3-none-any.whl (6.9 MB)
    |#####| 6.9 MB 3.1 MB/s
Collecting numpy>=1.14.0
  Downloading numpy-1.19.1-cp37-cp37m-win_amd64.whl (38.0 MB)
    |#####| 38.0 MB 22.0 MB/s
Collecting scipy<=1.4.1
  Downloading scipy-1.4.1-cp37-cp37m-win_amd64.whl (12.8 MB)
    |#####| 12.8 MB 3.2 MB/s
Collecting joblib<=0.11
  Downloading joblib-0.11.1-py2.py3-none-any.whl (294 kB)
    |#####| 294 kB 3.2 MB/s
Installing collected packages: numpy, scipy, joblib, scikit-learn
Successfully installed joblib-0.11.1 numpy-1.19.1 scikit-learn-0.22.1 scipy-1.4.1
(base) D:\VallanPython>cmd
  
```

**Gambar 1.1** Instalasi Scikit Learn

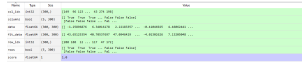
3. Setelah selesai instalasi, pilih salah satu example dari website Scikit (Contoh : )



**Gambar 1.2** Daftar Example



4. Lalu coba jalankan aplikasi tersebut, bisa dicek hasil dari Variable explorer



**Gambar 1.3** Variable Explorer

## 5. Sample kode

```

1 print(__doc__)
2
3 # Author: Kemal Eren <kemal@kemaleren.com>
4 # License: BSD 3 clause
5
6 import numpy as np
7 from matplotlib import pyplot as plt
8
9 from sklearn.datasets import make_biclusters
10 from sklearn.cluster import SpectralCoclustering
11 from sklearn.metrics import consensus_score
12
13 data, rows, columns = make_biclusters(
14     shape=(300, 300), n_clusters=5, noise=5,
15     shuffle=False, random_state=0)
16
17 plt.matshow(data, cmap=plt.cm.Blues)
18 plt.title("Original dataset")
19
20 # shuffle clusters
21 rng = np.random.RandomState(0)
22 row_idx = rng.permutation(data.shape[0])
23 col_idx = rng.permutation(data.shape[1])
24 data = data[row_idx][:, col_idx]
25
26 plt.matshow(data, cmap=plt.cm.Blues)
27 plt.title("Shuffled dataset")
28
29 model = SpectralCoclustering(n_clusters=5, random_state=0)
30 model.fit(data)
31 score = consensus_score(model.biclusters_,
32     (rows[:, row_idx], columns[:, col_idx]
33     ))
34
35 print("consensus score: {:.3f}".format(score))
36
37 fit_data = data[np.argsort(model.row_labels_)]
38 fit_data = fit_data[:, np.argsort(model.column_labels_)]
39
40 plt.matshow(fit_data, cmap=plt.cm.Blues)
41 plt.title("After biclustering; rearranged to show biclusters")

```

```
42 plt.show()
```

### 1.1.6 Mencoba Loading and example dataset

Disini akan dilakukan percobaan dengan menggunakan beberapa datasets seperti digits dan iris untuk bisa digunakan sebagai training set yang akan dipakai seluruh metode.

- Percobaan 1 (Memuat data iris dan digits dari datasets)

```
1 from sklearn import datasets #Untuk import class/fungsi
  dataset dari scikit-learn library
2 iris = datasets.load_iris() #Untuk memuat dan memasukkan
  dataset iris ke variabel bernama iris
3 digits = datasets.load_digits() #Untuk memuat dan memasukkan
  dataset digits ke variabel digits
```

```
In [18]: """
...: Created on Wed Feb 26 15:55:58 2020
...:
...: @author: Luthfi Muhammad Nabil
...: """
...:
...: from sklearn import datasets
...: iris = datasets.load_iris()
...: digits = datasets.load_digits()
```

**Gambar 1.4** Hasil Percobaan 1

- Percobaan 2 (Menampilkan data dari digits)

```
1 print(digits.data) #Menampilkan object berformat Dictionary-
  like yang nanti akan ditampilkan pada console
```

```
In [19]: print(digits.data)
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
```

**Gambar 1.5** Hasil Percobaan 2

- Percobaan 3 (Menampilkan digits.target)

```
1 digits.target #Menunjukkan data angka yang berhubungan dengan
  setiap digit gambar yang sedang dipelajari
```

```
In [20]: digits.target
Out[20]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.6 Hasil Percobaan 3

- Percobaan 4 (Menampilkan data 2 dimensi)

```
1 digits.images[0] #Akan mengambil data dengan berformat array
2 2 Dimensi, dengan bentuk parameter (n_samples, n_features)
```

```
In [22]: digits.images[0]
Out[22]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  0.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

Gambar 1.7 Hasil Percobaan 4

- Full sample

```
1 #-*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 15:55:58 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import datasets #Untuk import class/fungsi
9 dataset dari scikit-learn library
10 iris = datasets.load_iris() #Untuk memuat dan memasukkan
11 dataset iris ke variabel bernama iris
12 digits = datasets.load_digits() #Untuk memuat dan memasukkan
13 dataset digits ke variabel digits
14 #%%
15 print(digits.data) #Menampilkan object berformat Dictionary-
16 like yang nanti akan ditampilkan pada console
17 #%%
18 digits.target #Menunjukkan data angka yang berhubungan dengan
19 setiap digit gambar yang sedang dipelajari
20 #%%
21 digits.images[0] #Akan mengambil data dengan berformat array
22 2 Dimensi, dengan bentuk parameter (n_samples, n_features)
```



Gambar 1.8 Hasil pada variable explorer



```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:32:23 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import svm #Untuk mengimport class sv dari
   library sklearn
9 from sklearn import datasets #Untuk import class/fungsi
   dataset dari scikit-learn library
10 clf = svm.SVC(gamma=0.001, C=100) #Memasukkan implementasi
   dari "Support Vector Classification" ke variabel clf
11 iris = datasets.load_iris() #Untuk memuat dan memasukkan
   dataset iris ke variabel bernama iris
12
13 digits = datasets.load_digits() #Untuk memuat dan memasukkan
   dataset digits ke variabel digits
14 clf.fit(digits.data[:-1], digits.target[:-1]) #Untuk
   melakukan pengiriman data training set ke method fit
15
16 clf.predict(digits.data[-1:]) #Untuk melakukan prediksi nilai
   yang baru berdasarkan gambar terakhir dari digits.data

```



Gambar 1.12 Hasil pada variable explorer

## 1.1.8 Model Persistence

Disini akan dilakukan persistensi model menggunakan built-in dari Python

### ▪ Percobaan 1

```

1 from sklearn import svm #Mengimport class SVM dari library
   sklearn
2 from sklearn import datasets #Mengimport datasets dari
   library sklearn
3 clf = svm.SVC() #Memasukkan implementasi dari "Support Vector
   Classification" ke variabel clf
4 X, y = datasets.load_iris(return_X_y=True) #Untuk memuat dan
   memasukkan dataset iris ke variabel bernama iris
5 clf.fit(X, y) #Untuk melakukan pengiriman data training set
   ke method fit

```



Gambar 1.13 Hasil Percobaan 1

## ▪ Percobaan 2

```

1 import pickle #Mengimport pickle untuk implementasi protokol
  serializing dan de-serializing
2 s = pickle.dumps(clf) #Untuk serialize hirarki dari object
3 clf2 = pickle.loads(s) #Untuk memuat dan men de-serialize
  hirarki dari object
4 clf2.predict(X[0:1]) #Untuk melakukan prediksi nilai yang
  baru berdasarkan gambar terakhir dari digits.data

```

```

In [10]: import pickle #Mengimport pickle untuk implementasi protokol serializing dan de-serializing
Out[10]: <module 'pickle' from 'pickle.py'>
In [11]: s = pickle.dumps(clf) #Untuk serialize hirarki dari object
Out[11]: 'Pickle dump of the model'
In [12]: clf2 = pickle.loads(s) #Untuk memuat dan men de-serialize hirarki dari object
Out[12]: <model>
In [13]: clf2.predict(X[0:1]) #Untuk melakukan prediksi nilai yang baru berdasarkan gambar terakhir dari digits.data
Out[13]: array([0])

```

**Gambar 1.14** Hasil Percobaan 2

## ▪ Percobaan 3

```

1 y[0] #Untuk menampilkan data iris koordinat y

```

```

In [15]: y[0] #Untuk menampilkan data iris koordinat y
Out[15]: 0

```

**Gambar 1.15** Hasil Percobaan 3

## ▪ Percobaan 4

```

1 from joblib import dump, load #Mengambil class dump dan load
  dari library joblib
2 dump(clf, 'filename.joblib') #Untuk memasukkan data ke dalam
  sebuah file

```

```

In [16]: from joblib import dump, load #Mengambil class dump dan load dari library joblib
Out[16]: <module 'joblib' from 'joblib.py'>
In [17]: dump(clf, 'filename.joblib') #Untuk memasukkan data ke dalam sebuah file
Out[17]: 'filename.joblib'

```

**Gambar 1.16** Hasil Percobaan 4

## ▪ Percobaan 5

```

1 clf = load('filename.joblib') #Untuk memuat data dari file

```

```

In [17]: clf = load('filename.joblib') #Untuk memuat data dari file
Out[17]: <model>

```

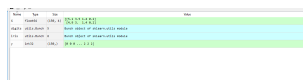
**Gambar 1.17** Hasil Percobaan 5

## ▪ Full sample

```

1  # -*- coding: utf-8 -*-
2  "" ""
3  Created on Thu Feb 27 10:37:49 2020
4
5  @author: Luthfi Muhammad Nabil
6  "" ""
7
8  from sklearn import svm #Mengimport class SVM dari library
   sklearn
9  from sklearn import datasets #Mengimport datasets dari
   library sklearn
10 clf = svm.SVC() #Memasukkan implementasi dari "Support Vector
   Classification" ke variabel clf
11 X, y = datasets.load_iris(return_X_y=True) #Untuk memuat dan
   memasukkan dataset iris ke variabel bernama iris
12 clf.fit(X, y) #Untuk melakukan pengiriman data training set
   ke method fit
13 #%%
14 import pickle #Mengimport pickle untuk implementasi protokol
   serializing dan de-serializing
15 s = pickle.dumps(clf) #Untuk serialize hirarki dari object
16 clf2 = pickle.loads(s) #Untuk memuat dan men de-serialize
   hirarki dari object
17 clf2.predict(X[0:1]) #Untuk melakukan prediksi nilai yang
   baru berdasarkan gambar terakhir dari digits.data
18 #%%
19 y[0] #Untuk menampilkan data iris koordinat y
20 #%%
21 from joblib import dump, load #Mengambil class dump dan load
   dari library joblib
22 dump(clf, 'filename.joblib') #Untuk memasukkan data ke dalam
   sebuah file
23 #%%
24 clf = load('filename.joblib') #Untuk memuat data dari file

```



Variable	Type	Value
clf	SVC	SVC (C=1.0, gamma=0.5)
clf2	SVC	SVC (C=1.0, gamma=0.5)

Gambar 1.18 Hasil pada variable explorer

## 1.1.9 Conventions

Seluruh metode akan dilakukan pengaturan untuk membuat tingkah laku lebih dapat diprediksi

- Percobaan 1

```

1 import numpy as np #Memuat library numpy yang akan
   diinisialisasikan menjadi np
2 from sklearn import random_projection #Memuat class
   random_projection dari library sklearn

```

3





- Percobaan 4

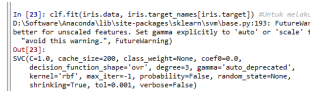
```
1 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan
    dimasukkan ke fungsi list
```



Gambar 1.22 Hasil Percobaan 4

- Percobaan 5

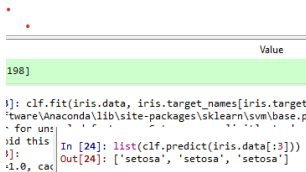
```
1 clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
    melakukan pengiriman data training set ke method fit
```



Gambar 1.23 Hasil Percobaan 5

- Percobaan 6

```
1 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan
    dimasukkan ke fungsi list
```



Gambar 1.24 Hasil Percobaan 6

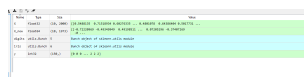
- Full sample

```
1 # -*- coding: utf-8 -*-
2 "" ""
3 Created on Thu Feb 27 10:56:18 2020
4
5 @author: Luthfi Muhammad Nabil
```

```

6  %% %%
7
8  import numpy as np #Memuat library numpy yang akan
   diinisialisasikan menjadi np
9  from sklearn import random_projection #Memuat class
   random_projection dari library sklearn
10
11  rng = np.random.RandomState(0) #Memuat angka random dan
   mengekspos angka untuk menghasilkan dari berbagai
   probabilitas distribusi, angka tersebut dimasukkan ke
   variabel rng
12  X = rng.rand(10, 2000) #Membatasi jarak angka random
13  X = np.array(X, dtype='float32') #Memuat angka menjadi ke
   dalam array
14  X.dtype #Menggambil tipe data dari variabel X
15  #%%
16  transformer = random_projection.GaussianRandomProjection() #
   Mengimplementasikan modul yang simpel dan efisien secara
   komputasi untuk mengurangi dimensi dari data
17  X_new = transformer.fit.transform(X) #Untuk membuat
   penengahan data
18  X_new.dtype #Menggambil tipe data dari variabel X
19
20  #%%
21  from sklearn import datasets #Mengimport datasets dari
   library sklearn
22  from sklearn.svm import SVC
23  iris = datasets.load_iris() #Untuk memuat dan memasukkan
   dataset iris ke variabel bernama iris
24  clf = SVC() #Memasukkan implementasi dari "Support Vector
   Classification" ke variabel clf
25  clf.fit(iris.data, iris.target) #Untuk melakukan pengiriman
   data training set ke method fit
26  #%%
27  list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
   serialize hirarki dari object, data tersebut akan
   dimasukkan ke fungsi list
28  #%%
29  clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
   melakukan pengiriman data training set ke method fit
30  #%%
31  list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
   serialize hirarki dari object, data tersebut akan
   dimasukkan ke fungsi list

```



**Gambar 1.25** Hasil pada variable explorer

## 1.1.10 Skrinsut Error

```

1: File "C:\Users\Luthfi\Documents\RandomProjection.py", line 14, in <module>
2:     random = np.random.RandomState(0)
3: NameError: name 'random' is not defined
4:
5: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
6:   random = np.random.RandomState(0)
7:
8: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
9:   random = np.random.RandomState(0)
10:
11: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
12:   random = np.random.RandomState(0)
13:
14: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
15:   random = np.random.RandomState(0)
16:
17: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
18:   random = np.random.RandomState(0)
19:
20: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
21:   random = np.random.RandomState(0)
22:
23: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
24:   random = np.random.RandomState(0)
25:
26: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
27:   random = np.random.RandomState(0)
28:
29: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
30:   random = np.random.RandomState(0)
31:
32: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
33:   random = np.random.RandomState(0)
34:
35: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
36:   random = np.random.RandomState(0)
37:
38: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
39:   random = np.random.RandomState(0)
40:
41: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
42:   random = np.random.RandomState(0)
43:
44: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
45:   random = np.random.RandomState(0)
46:
47: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
48:   random = np.random.RandomState(0)
49:
50: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
51:   random = np.random.RandomState(0)
52:
53: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
54:   random = np.random.RandomState(0)
55:
56: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
57:   random = np.random.RandomState(0)
58:
59: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
60:   random = np.random.RandomState(0)
61:
62: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
63:   random = np.random.RandomState(0)
64:
65: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
66:   random = np.random.RandomState(0)
67:
68: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
69:   random = np.random.RandomState(0)
70:
71: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
72:   random = np.random.RandomState(0)
73:
74: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
75:   random = np.random.RandomState(0)
76:
77: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
78:   random = np.random.RandomState(0)
79:
80: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
81:   random = np.random.RandomState(0)
82:
83: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
84:   random = np.random.RandomState(0)
85:
86: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
87:   random = np.random.RandomState(0)
88:
89: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
90:   random = np.random.RandomState(0)
91:
92: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
93:   random = np.random.RandomState(0)
94:
95: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
96:   random = np.random.RandomState(0)
97:
98: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined
99:   random = np.random.RandomState(0)
100: C:\Users\Luthfi\Documents\RandomProjection.py:14: NameError: name 'random' is not defined

```

Gambar 1.26 Hasil Percobaan 6

### 1.1.11 Kode error dan jenis error tersebut

Error yang didapat berjenis name error, karena sebuah variabel tidak didefinisikan. Yaitu digits

```

1  # -*- coding: utf-8 -*-
2  "" ""
3  Created on Thu Feb 27 10:56:18 2020
4
5  @author: Luthfi Muhammad Nabil
6  "" ""
7
8  import numpy as np #Memuat library numpy yang akan
9  diinisialisasikan menjadi np
10 from sklearn import random_projection #Memuat class
11 random_projection dari library sklearn
12
13 rng = np.random.RandomState(0) #Memuat angka random dan
14 mengekspos angka untuk menghasilkan dari berbagai
15 probabilitas distribusi, angka tersebut dimasukkan ke
16 variabel rng
17 X = rng.rand(10, 2000) #Membatasi jarak angka random
18 X = np.array(X, dtype='float32') #Memuat angka menjadi ke dalam
19 array
20 X.dtype #Mengambil tipe data dari variabel X
21 #%%
22 transformer = random_projection.GaussianRandomProjection() #
23 Mengimplementasikan modul yang simpel dan efisien secara
24 komputasi untuk mengurangi dimensi dari data
25 X_new = transformer.fit_transform(X) #Untuk membuat penengahan
26 data
27 X_new.dtype #Mengambil tipe data dari variabel X
28
29 #%%
30 from sklearn import datasets #Mengimport datasets dari library
31 sklearn
32 from sklearn.svm import SVC
33 iris = datasets.load_iris() #Untuk memuat dan memasukkan dataset
34 iris ke variabel bernama iris
35 clf = SVC() #Memasukkan implementasi dari "Support Vector
36 Classification" ke variabel clf
37 clf.fit(iris.data, iris.target) #Untuk melakukan pengiriman data
38 training set ke method fit
39
40 #%%

```

```

27 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan dimasukkan
    ke fungsi list
28 ###
29 clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
    melakukan pengiriman data training set ke method fit
30 ###
31 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan dimasukkan
    ke fungsi list

```

### 1.1.12 Penanganan Error

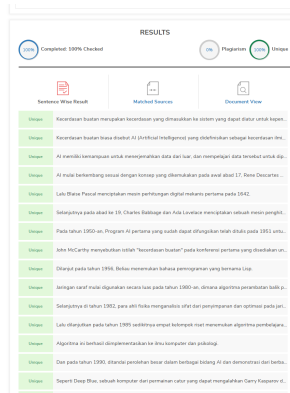
Untuk menangani error tersebut, bisa ditambahkan sesuai instruksi. Yaitu menambahkan sebuah variabel bernama digits. Selain itu, digits harus dapat bekerja sebagaimana mestinya. Berikut full kodingnya :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:32:23 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import svm #Untuk mengimport class sv dari library
    sklearn
9 from sklearn import datasets #Untuk import class/fungsi dataset
    dari scikit-learn library
10 clf = svm.SVC(gamma=0.001, C=100) #Memasukkan implementasi dari "
    Support Vector Classification" ke variabel clf
11 iris = datasets.load_iris() #Untuk memuat dan memasukkan dataset
    iris ke variabel bernama iris
12 ###
13 digits = datasets.load_digits() #Untuk memuat dan memasukkan
    dataset digits ke variabel digits
14 clf.fit(digits.data[:-1], digits.target[:-1]) #Untuk melakukan
    pengiriman data training set ke method fit
15 ###
16 clf.predict(digits.data[-1:]) #Untuk melakukan prediksi nilai
    yang baru berdasarkan gambar terakhir dari digits.data

```

## 1.1.13 Plagiarisme



Gambar 1.27 Hasil pada variable explorer

## 1.2 1174040 - Hagan Rowlenstino A. S

### 1.2.1 Teori

**1.2.1.1 Definisi Kecerdasan Buatan** Artificial Intelligence atau dapat disebut juga dengan kecerdasan buatan merupakan kecerdasan yang ditambahkan kepada suatu sistem yang dapat diatur dalam konteks ilmiah. Michael Haenlein dan Andreas Kaplan mendefinisikan bahwa AI adalah “sistem yang mempunyai kemampuan untuk menguraikan, belajar agar dapat digunakan untuk mencapai tujuan dengan melalui adaptasi yang fleksibel”. Kecerdasan ini dibuat dan dimasukkan ke dalam mesin agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia dengan cepat dan tepat. Bidang-bidang yang menggunakan kecerdasan buatan antara lain logika fuzzy, games, sistem pakar, jaringan saraf tiruan dan robotika.

**1.2.1.2 Sejarah Kecerdasan Buatan** Sejarah kecerdasan buatan ini dimulai dari zaman kuno, mitos ataupun cerita dan desas-desus tentang sebuah makhluk yang mempunyai kecerdasan serta kesadaran yang diberikan oleh pengrajin. Benih - benih nya mulai ditanam oleh para filsuf klasik yang mencari cara untuk menggambarkan proses berfikir manusia sebagai manipulasi simbol secara mekanis yang memuncak pada penemuan komputer digital di tahun 1940-an, yaitu sebuah mesin yang didasarkan penalaran matematika. Istilah kecerdasan buatan sendiri baru muncul pada tahun 1956, dan teori-teori nya sudah muncul sejak tahun 1941.

### 1.2.1.3 Perkembangan Kecerdasan Buatan

1. Perkembangan kecerdasan buatan dimulai dari Era Komputer Elektronik pada tahun 1941. dimana ditemukannya alat penyimpanan dan pemros-

esan informasi. Dilanjutkan pada tahun 1949, berhasilnya pembuatan komputer yang mampu menyimpan program yang memuat pekerjaan dalam memasukkan program menjadi lebih mudah.

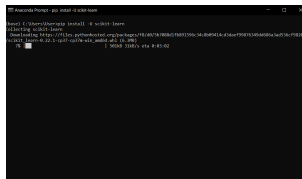
2. Masa - masa persiapan AI terjadi pada tahun 1943 - 1956.
3. Awal Perkembangan AI terjadi pada 1952 - 1969
4. Perkembagan kecerdasan buatan melambat pada tahun 1966-1974
5. Sistem Berbasis Pengetahuan pada tahun 1969-1979
6. ecerdasan Buatan menjadi sebuah industri pada tahun 1980-1988. Kembalinya Jaringan Syaraf tiruan pada tahun 1986-sekarang.

## 1.2.2 Instalasi

Membuka <https://scikit-learn.org/stable/tutorial/basic/tutorial.html> lalu mencobanya.

### 1.2.2.1 Instalasi library scikit, mencoba kompilasi dan ujicoba contoh kode

1. Buka anaconda prompt lalu ketikkan "pip install -U scikit-learn" untuk menginstall library scikit



**Gambar 1.28** Install Library Scikit

2. Pilih salah satu example dari website tersebut lalu jalankan

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 18:16:44 2020
4
5 @author: User
6 """
7
8 import matplotlib.pyplot as plt
9 from sklearn.svm import SVC
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.metrics import plot_roc_curve
12 from sklearn.datasets import load_wine
13 from sklearn.model_selection import train_test_split

```

```

14
15 X, y = load_wine(return_X_y=True)
16 y = y == 2
17
18 X_train, X_test, y_train, y_test = train_test_split(X, y,
19 random_state=42)
20 svc.fit(X_train, y_train)

```

### 3. buka variable explolernya

Name	Type	Size	Value
X	float64	(178, 11)	[1.422e+01 1.710e+00 2.430e+00 ... 1.848e+00 1.920e+00]
X_test	float64	(45, 11)	[1.145e+01 1.510e+00 2.450e+00 ... 1.980e+00 2.450e+00]
X_train	float64	(133, 11)	[1.145e+01 1.510e+00 2.450e+00 ... 1.980e+00 2.450e+00]
y	bool	(178,)	[1 ...]
y_test	bool	(45,)	[False False True ... True True True]
y_train	bool	(133,)	[False False True ... False False True]

Gambar 1.29 Variable Explorer

#### 1.2.2.2 Mencoba loading an example dataset

##### 1. mengambil data iris dan digit dari dataset

```

1 from sklearn import datasets #untuk mengimport dataset dari
   library learn-scikit
2 iris = datasets.load_iris() #membuat sebuah variable iris
   yang mempunyai isi yaitu dataset iris
3 digits = datasets.load_digits() #membuat sebuah variable
   digits yang mempunyai isi yaitu dataset digits

```

##### 2. Menampilkan data digits

```

1 print(digits.data) #memberikan akses ke fitur yang dapat
   digunakan untuk mengklasifikasikan sampel digit dan
   menampilkannya di console

```

```

In [18]: runfile('D:/Semester/Al/Chapter1/m2.py', wdir='D:/Semester/Al/Chapter1')
Out[18]:
[[ 0.  0. ...  0.  0.]
 [ 0.  0. ... 10.  0.]
 [ 0.  0. ... 16.  0.]
 ...
 [ 0.  0. ...  0.  0.]
 [ 0.  0. ... 12.  0.]
 [ 0.  0. ... 12.  1.  0.]]

```

Gambar 1.30 Data Digits

##### 3. menampilkan digits.target

```

1 digits.target #memberikan informasi tentang data yang
   berhubungan atau juga dapat dijadikan sebagai label

```

```
In [16]: digits.target
Out[16]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.31 Digits Target

#### 4. menampilkan data bentuk 2D.

```
1 digits.images[0] #Data selalu berupa array 2D, shape (
    n_samples, n_features), meskipun data aslinya mungkin
    memiliki bentuk yang berbeda.
```

```
In [17]: digits.images[0]
Out[17]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

Gambar 1.32 Data 2D

### 1.2.2.3 Mencoba Learning and Predicting

```
1 #-*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 22:20:55 2020
4
5 @author: User
6 """
7
8 from sklearn.linear_model import LogisticRegression #untuk
    mengimport linear_model dari library sklearn
9 from sklearn.datasets import make_blobs #untuk mengimport library
    datasets dari sklearn
10
11 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
    random_state=1) # untuk generate dataset dengan klasifikasi 2
    D
12 model = LogisticRegression() #menggunakan metode loginstic
    regression
13 model.fit(X, y)
14
15 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
    random_state=1) # menentukan 1 buah contoh baru dimana
    jawabannya tidak diketahui
16
17 ynew = model.predict_proba(Xnew) # membuat sebuah prediksi dan
    memasukkan nya kedalam variable ynew
18 for i in range(len(Xnew)):
19     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #menampilkan
    hasil prediksi
```



### 1.2.2.4 Mencoba Model Persistence

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 22:27:13 2020
4
5 @author: User
6 """
7
8 from sklearn import svm #menigimport svm dari library sklearn
9 from sklearn import datasets #menigimport datasets dari library
   sklearn
10 clf = svm.SVC() #menggunakan method SVC
11 iris = datasets.load_iris() #menggunakan dataset iris
12 X, y = iris.data, iris.target #memasukkan x sebagai iris data ,
   dan y sebagai iris target
13 clf.fit(X, y) #laalu menggunakan metod fit .

```

### 1.2.2.5 Mencoba Conventions

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 23:24:21 2020
4
5 @author: User
6 """
7
8 import numpy as np #mengimport numpy sebagai npy
9 from sklearn import random_projection #mengimport
   random_projection dari library sklearn
10
11 rng = np.random.RandomState(0) #Menggunakan fungsi random dari
   numpy
12 X = rng.rand(10, 2000) #membuat range random diantara 10 sampai
   2000
13 X = np.array(X, dtype='float32') #yang dijadikan array dengan
   tipe data float32
14 X.dtype

```

## 1.2.3 Penanganan Error

```

context))
ValueError: Found array with 8 sample(s) (shape=(8, 2)) while a minimum of 1 is
required.

```

**Gambar 1.33** Data 2D

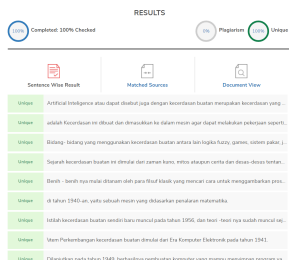
### 1.2.3.1 Screenshot Error

### 1.2.3.2 Kode error dan jenis error Jenis errornya adalah value error

```
1 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
    random_state=1) # menentukan 1 buah contoh baru dimana
    jawabannya tidak diketahui
```

**1.2.3.3 Solusi Error** Solusinya adalah dengan mengganti nilai `n_samples` nya agar tidak 0

## 1.2.4 Cek Plagiarism



**Gambar 1.34** Cek Plagiarism

## 1.3 1174042 Faisal Najib Abdullah

### 1.3.1 Teori

1. Definisi, sejarah, dan perkembangan kecerdasan buatan.

Definisi kecerdasan buatan adalah suatu pengetahuan yang dapat membuat komputer untuk meniru kecerdasan manusia yang berhubungan dengan penangkapan, pemodelan, dan penyimpanan kecerdasan manusia dalam sebuah sistem teknologi. Contohnya yaitu melakukan analisa penalaran untuk mengambil suatu kesimpulan atau penerjemahan atau keputusan dari satu bahasa satu ke bahasa lain.

Sejarah dan perkembangan kecerdasan buatan terjadi pada musim panas tahun 1956 tercatat adanya seminar mengenai AI di Darmouth College. Seminar pada waktu itu dihadiri oleh sejumlah pakar komputer dan membahas potensi komputer dalam meniru kepandaian manusia. Akan tetapi perkembangan yang sering terjadi semenjak diciptakannya LISP, yaitu bahasa kecerdasan buatan yang dibuat tahun 1960 oleh John McCarthy. Istilah pada kecerdasan buatan atau Artificial Intelligence diambil dari Marvin Minsky dari MIT. Dia menulis karya ilmiah berjudul *Step towards Artificial Intelligence*, The Institute of radio Engineers Proceedings 49, January 1961[?].

2. Definisi supervised learning, klasifikasi, regresi, dan unsupervised learning. Data set, training set dan testing set.

Supervised learning merupakan sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokkan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokkan data tersebut menjadi 2 bagian atau 3 bagian dan seterusnya.

Klasifikasi adalah salah satu topik utama dalam data mining atau machine learning. Klasifikasi yaitu suatu pengelompokan data dimana data yang digunakan tersebut mempunyai kelas label atau target.

Regresi adalah Supervised learning tidak hanya mempelajari classifier, tetapi juga mempelajari fungsi yang dapat memprediksi suatu nilai numerik. Contoh, ketika diberi foto seseorang, kita ingin memprediksi umur, tinggi, dan berat orang yang ada pada foto tersebut.

Data set adalah cabang aplikasi dari Artificial Intelligence/Kecerdasan Buatan yang fokus pada pengembangan sebuah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia.

Training set yaitu jika pasangan objek, dan kelas yang menunjuk pada objek tersebut adalah suatu contoh yang telah diberi label akan menghasilkan suatu algoritma pembelajaran.

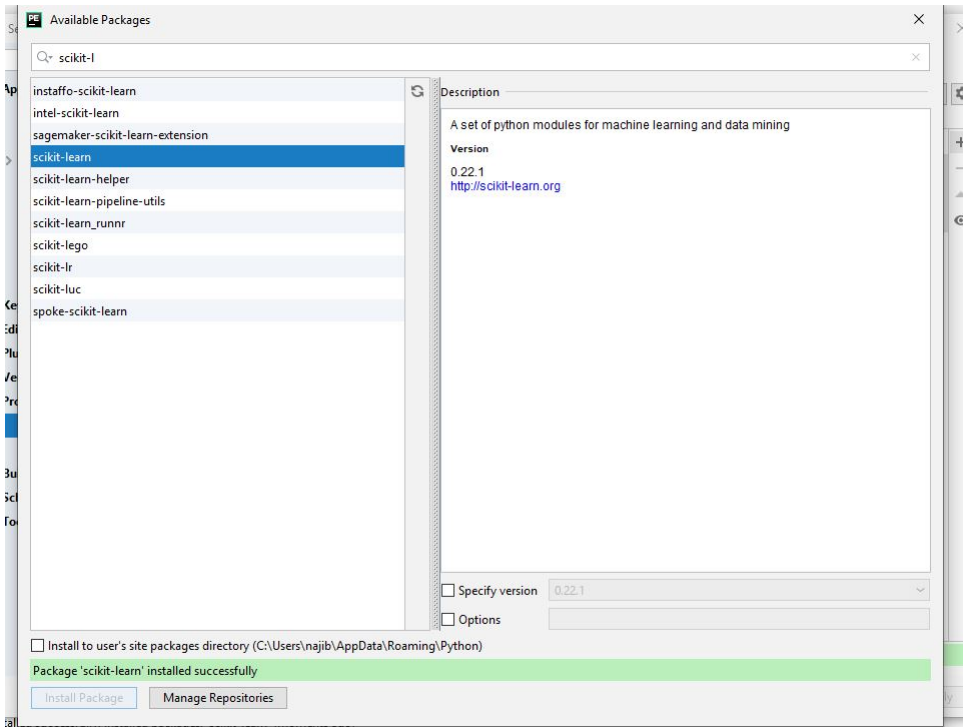
Testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar[?].

## 1.3.2 Instalasi

**1.3.2.1 Instalasi Library Scikit dari Pycharm** Masuk pada menu settings terus pilih Project Interpreter kemudian tambah library lalu cari dan install scikit Mencoba Library

```
1 from sklearn import datasets
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
  datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #pada baris kedua ini dimana iris merupakan suatu estimator/
  parameter yang berfungsi untuk mengambil data pada item
  datasets.load_iris
5 digits = datasets.load_digits()
6 #pada baris ketiga ini dimana digits merupakan suatu estimator/
  parameter yang berfungsi untuk mengambil data pada item
  datasets.load_digits
```

```
1 from sklearn import datasets
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
  datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #pada baris kedua ini dimana iris merupakan suatu estimator/
  parameter yang berfungsi untuk mengambil data pada item
  datasets.load_iris
```



Gambar 1.35 Instalasi

```

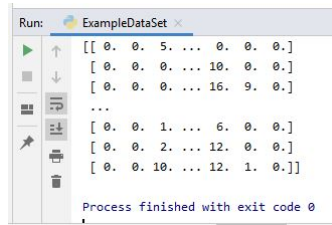
5 digits = datasets.load_digits()
6 #pada baris ketiga ini dimana digits merupakan suatu estimator/
  parameter yang berfungsi untuk mengambil data pada item
  datasets.load_digits
7 print(digits.data)
8 #pada baris keempat ini merupakan perintah yang berfungsi untuk
  menampilkan estimator/parameter yang dipanggil pada item
  digits.data dan menampilkan outputannya
9 digits.target
10 #barisan ini untuk mengambil target pada estimator/parameter
   digits dan menampilkan outputannya
11 digits.images[0]
12 #barisan ini untuk mengambil images[0] pada estimator/parameter
   digits dan menampilkan outputannya

```

```

1 from sklearn import svm
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
  svm dari packaged sklearn
3 clf = svm.SVC(gamma=0.001, C=100.)

```



**Gambar 1.36** Mencoba Loading an example Dataset

```

4 #pada baris kedua ini clf sebagai estimator/parameter, svm.SVC
  sebagai class, gamma sebagai parameter untuk menetapkan nilai
  secara manual
5 clf.fit(digits.data[:-1], digits.target[:-1])
6 #pada baris ketiga ini clf sebagai estimator/parameter, fit
  sebagai metode, digits.data sebagai item, [:-1] sebagai
  syntax pythonnya dan menampilkan outputnya
7 clf.predict(digits.data[:-1])
8 #pada baris terakhir ini clf sebagai estimator/parameter, predict
  sebagai metode lainnya, digits.data sebagai item dan
  menampilkan outputnya

```

```

Terminal: Local +
>>> clf = svm.SVC(gamma=0.001, C=100.)
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
>>> clf.predict(digits.data[:-1])
array([8])
>>>

```

**Gambar 1.37** Learning and Predicting

```

1 from sklearn import svm
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
  svm dari packaged sklearn
3 from sklearn import datasets
4 #pada baris ini merupakan sebuah perintah untuk mengimport class
  datasets dari packaged sklearn
5 clf = svm.SVC(gamma='scale')
6 #pada baris ketiga ini clf sebagai estimator/parameter, svm.SVC
  sebagai class, gamma sebagai parameter untuk menetapkan nilai
  secara manual dengan nilai scale
7 iris = datasets.load_iris()
8 #pada baris keempat ini iris sebagai estimator/parameter,
  datasets.load_iris() sebagai item dari suatu nilai
9 X, y = iris.data, iris.target
10 #pada baris kelima ini X, y sebagai estimator/parameter, iris.
    data, iris.target sebagai item dari 2 nilai yang ada

```

```

11 clf.fit(X, y)
12 #pada baris keenam ini clf sebagai estimator/parameter dengan
    menggunakan metode fit untuk memanggil estimator X, y dengan
    outputannya
13
14
15 import pickle
16 #pickle merupakan sebuah class yang di import
17 s = pickle.dumps(clf)
18 #pada baris ini s sebagai estimator/parameter dengan pickle.dumps
    merupakan suatu nilai/item dari estimator/parameter clf
19 clf2 = pickle.loads(s)
20 #pada baris ini clf2 sebagai estimator/parameter, pickle.loads
    sebagai suatu item, dan s sebagai estimator/parameter yang
    dipanggil
21 clf2.predict(X[0:1])
22 #pada baris ini clf2.predict sebagai suatu item dengan
    menggunakan metode predict untuk menentukan suatu nilai dari
    (X[0:1])
23 y[0]
24 #pada estimator/parameter y berapapun angka yang diganti nilainya
    akan selalu konstan yaitu 0
25
26
27 from joblib import dump, load
28 #pada baris berikut ini merupakan sebuah perintah untuk
    mengimport class dump, load dari packaged joblib
29 dump(clf, 'filename.joblib')
30 #pada baris berikutnya dump di sini sebagai class yang didalamnya
    terdapat nilai dari suatu item clf dan data joblib
31 clf = load('filename.joblib')
32 #pada baris terakhir clf sebagai estimato/parameter dengan suatu
    nilai load berfungsi untuk mengulang data sebelumnya

```

```

... clf.fit(X, y)
SVC(C=100.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)

```

Gambar 1.38 Model Presistence

```

>>> clf2.predict(X[0:1])
array([0])
>>> #pada baris ini clf2.predict
... y[0]
0

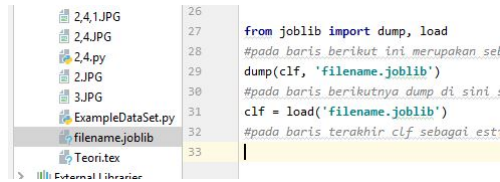
```

Gambar 1.39 Model Presistence

```

1 #Type Casting
2 from sklearn import svm

```



Gambar 1.40 Model Persistence

```

3 #pada baris ini merupakan sebuah perintah untuk mengimport class
  svm dari packaged sklearn
4 from sklearn import random_projection
5 #pada baris ini merupakan sebuah perintah untuk mengimport class
  random_projection dari packaged sklearn
6 rng = np.random.RandomState(0)
7 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
  np.random.RandomState(0)
8 X = rng.rand(10, 2000)
9 #X sebagai estimator/parameter dengan nilai item rng.rand
10 X = np.array(X, dtype='float32')
11 #X sebagai estimator/parameter dengan nilai item np.array
12 X.dtype
13 #X.dtype sebagai item pemanggil
14 transformer = random_projection.GaussianRandomProjection()
15 #transformer sebagai estimator/parameter dengan memanggil class
  random_projection
16 X_new = transformer.fit_transform(X)
17 #X_new di sini sebagai estimator/parameter dan menggunakan metode
  fit
18 X_new.dtype
19 #X_new.dtype sebagai item
20
21
22 from sklearn import datasets
23 #pada baris ini merupakan sebuah perintah untuk mengimport class
  datasets dari packaged sklearn
24 from sklearn.svm import SVC
25 #pada baris ini merupakan sebuah perintah untuk mengimport class
  SVC dari packaged sklearn.svm
26 iris = datasets.load_iris()
27 #iris sebagai estimator/parameter dengan item datasets.load_iris
  ()
28 clf = SVC(gamma='scale')
29 #clf sebagai estimator/parameter dengan nilai class SVC pada
  parameter gamma sebagai set penilaian
30 clf.fit(iris.data, iris.target)
31 #estimator/parameter clf menggunakan metode fit dengan itemnya
32 list(clf.predict(iris.data[:3]))
33 #menambahkan item list dengan metode predict
34 clf.fit(iris.data, iris.target_names[iris.target])
35 #estimator/parameter clf menggunakan metode fit dengan itemnya
36 list(clf.predict(iris.data[:3]))
37 #menambahkan item list dengan metode predict
38

```

```

39
40
41 #Refitting and Updating Parameters
42 import numpy as np
43 #pada baris ini merupakan sebuah perintah untuk mengimport class
   svm dari np
44 from sklearn.svm import SVC
45 #pada baris ini merupakan sebuah perintah untuk mengimport class
   SVC dari packaged sklearn.svm
46 rng = np.random.RandomState(0)
47 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
   np.random.RandomState(0)
48 X = rng.rand(100, 10)
49 #X sebagai estimator/parameter dengan nilai item rng.rand
50 y = rng.binomial(1, 0.5, 100)
51 #y sebagai estimator/parameter dengan nilai item rng.binomial
52 X_test = rng.rand(5, 10)
53 #X test sebagai estimator/parameter dengan nilai item rng.rand
54 clf = SVC()
55 #clf sebagai estimator/parameter dan class SVC
56 clf.set_params(kernel='linear').fit(X, y)
57 #set params sebagai item
58 clf.predict(X_test)
59 #menggunakan metode predict
60 clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
61 clf.predict(X_test)
62
63
64 #Multiclass vs. Multilabel Fitting
65 from sklearn.svm import SVC
66 #pada baris ini merupakan sebuah perintah untuk mengimport class
   SVC dari packaged sklearn.svm
67 from sklearn.multiclass import OneVsRestClassifier
68 #pada baris ini merupakan sebuah perintah untuk mengimport class
   OneVsRestClassifier dari packaged sklearn.multiclass
69 from sklearn.preprocessing import LabelBinarizer
70 #pada baris ini merupakan sebuah perintah untuk mengimport class
   LabelBinarizer dari packaged sklearn.preprocessing
71 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
72 y = [0, 0, 1, 1, 2]
73 classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
   random_state=0))
74 classif.fit(X, y).predict(X)
75 y = LabelBinarizer().fit_transform(y)
76 classif.fit(X, y).predict(X)
77
78
79 from sklearn.preprocessing import MultiLabelBinarizer
80 y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
81 y = MultiLabelBinarizer().fit_transform(y)
82 classif.fit(X, y).predict(X)

```

### 1.3.3 Penanganan error



```
>>> from joblib import dump, load
>>> #pada baris berikut ini merupakan sebuah perintah untuk mengimpor class dump, load dari packaged joblib
... dump(clf, 'filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
NameError: name 'clf' is not defined
>>> #pada baris berikutnya dump di sini sebagai class yang didalamnya terdapat nilai dari suatu item clf dan data joblib
... clf = load('filename.joblib')
```

Gambar 1.41 Error

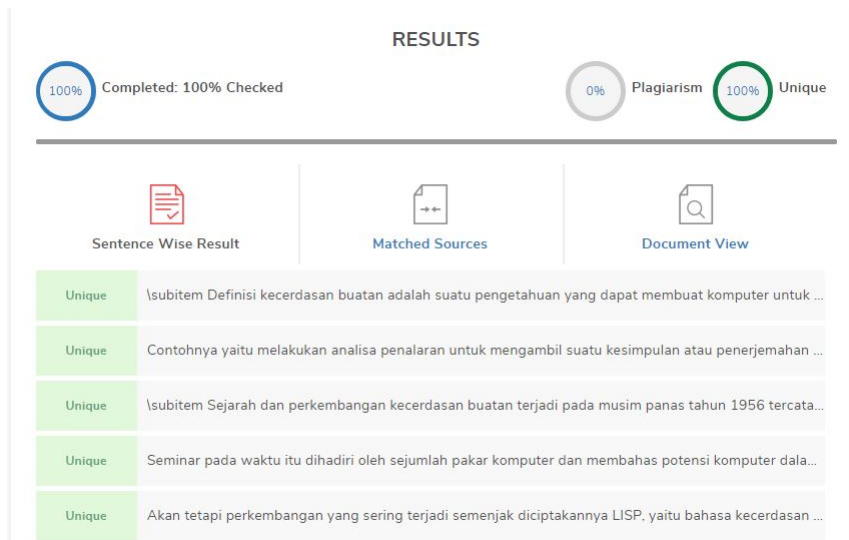
### 1.3.3.1 ScreenShoot Error

### 1.3.3.2 Tuliskan Kode Error dan Jenis Erornya

File "<stdin>", line 2, in <module>  
NameError: name 'clf' is not defined

**1.3.3.3 Solusi Pemecahan Masalah Error** Ini karna kode di jalankan perbaris perbaris, jika kode dijanlankan bersamaan makan kode berjan sesuai prosedur.

## 1.3.4 Plagiat



Gambar 1.42 Error

## 1.3.5 Link

[Youtube](#)

## 1.4 1174043 - Irvan Rizkiansyah

### 1.4.1 Definisi Kecerdasan Buatan

Kecerdasan buatan atau biasa disebut AI (Artificial Intelligence) merupakan kecerdasan yang dibuat dan ditambahkan oleh manusia ke suatu sistem teknologi, yang diatur dan dikembangkan dalam konteks ilmiah, yang merupakan bentuk dari kecerdasan entitas ilmiah yang ada. Jadi pada intinya definisi AI dapat terus dikembangkan, namun yang menjadi poin utamanya adalah bagaimana manusia menciptakan sebuah teknologi yang dapat berpikir seperti selayaknya manusia itu sendiri.

### 1.4.2 Sejarah dan Perkembangan

- Program kecerdasan buatan atau Artificial Intelligence pertama kali diceptuskan pada kisaran tahun 1951. Tidak bisa dipungkiri bahwa di tahun tersebut memang sedang gencar-gencarnya pembuatan cikal bakal, konsep, hingga teknologi berbasis AI. Dan, AI sendiri pertama kali digunakan di University of Manchester untuk menjalankan sebuah mesin bernama Ferranti Mark 1.
- Beberapa waktu kemudian, Christopher Strachey melanjutkan konsep kecerdasan buatan untuk menjalankan sebuah permainan catur, dimana bidak catur tersebut dapat berjalan secara otomatis dan mampu bermain melawan manusia sungguhan.
- Berlanjut pada tahun 1956, kecerdasan buatan tidak hanya dibuat untuk memudahkan bermain catur saja. Melainkan pada saat konferensi pertamanya, John McCharty menamai algoritma teknologi tersebut dengan sebutan “Artificial Intelligence”. Istilah tersebut masih digunakan hingga sekarang oleh para pakar teknologi.
- Terakhir, konsep dan teknologi kecerdasan buatan disempurnakan oleh seorang ahli yang namanya masih diingat sampai sekarang sebagai seorang pakar kecerdasan buatan, yaitu Alan Turin. Pada saat itu, Alan Turin meneliti dan menguji coba algoritma AI yang diberi nama dengan “Turing Test”. Hingga seiring berkembangnya waktu, konsep teknologi AI banyak digunakan di berbagai teknologi baik itu multimedia, search engine, dan masih banyak lainnya.

### 1.4.3 Supervised Learning

Supervised learning (Pembelajaran terawasi), dalam konteks kecerdasan buatan (AI) dan Machine Learning, adalah jenis sistem di mana input dan output data yang diinginkan disediakan. Input dan output data diberi label

untuk klasifikasi untuk memberikan dasar pembelajaran untuk pemrosesan data di masa depan.

#### 1.4.4 Unsupervised Learning

Unsupervised learning merupakan pembelajaran yang tidak terawasi dimana tidak memerlukan target output. Pada metode ini tidak dapat ditentukan hasil seperti apa yang diharapkan selama proses pembelajaran, nilai bobot yang disusun dalam proses range tertentu tergantung pada nilai output yang diberikan. Tujuan metode unsupervised learning ini agar kita dapat mengelompokkan unit-unit yang hampir sama dalam satu area tertentu.

#### 1.4.5 Teknik Klasifikasi

Teknik klasifikasi memprediksi respons diskrit, misalnya seperti apakah email itu asli atau spam, atau apakah tumor itu kanker atau tidak. Model klasifikasi mengklasifikasikan data masukan ke dalam kategori tersebut.

#### 1.4.6 Regresi

Regresi yaitu pengeluaran nilai output yang konstan jika dipicu dengan parameter tertentu biasanya regresi disini berbentuk regresi linier. Regresi linier yaitu metode statistika yang digunakan untuk membentuk model hubungan antara variabel terikat(dependen, respon,  $Y$ ) dengan satu atau lebih variabel bebas(independent, prediktor,  $X$ ). Apabila banyaknya variabel bebas hanya ada satu, disebut sebagai regresi linier sederhana, sedangkan apabila terdapat lebih dari satu variabel bebas, disebut sebagai regresi linier berganda.

#### 1.4.7 Training Set

Training set adalah bagian dari dataset itu sendiri yang dilatih untuk membuat prediksi atau algoritma mesin learning lainnya sesuai keinginan atau tujuan data itu dibuat.

#### 1.4.8 Testing Set

Testing set adalah bagian dari dataset yang di tes atau diujicoba untuk melihat keakuratannya dengan katalain melihat peformanya.

#### 1.4.9 Instalasi dan Percobaan Kompilasi dari Library Scikit-learn

1. Buka anaconda prompt
2. kemudian Ketik di anaconda prompt yaitu : "pip install -U scikit-learn"



Gambar 1.43 Instalasi Scikit Learn

3. Setelah selesai instalasi, pilih salah satu example dari website Scikit
4. Sample kode

```

1 print(__doc__)
2
3 # Author: Nelle Varoquaux <nelle.varoquaux@gmail.com>
4 #         Alexandre Gramfort <alexandre.gramfort@inria.fr>
5 # License: BSD
6
7 import numpy as np
8 import matplotlib.pyplot as plt
9 from matplotlib.collections import LineCollection
10
11 from sklearn.linear_model import LinearRegression
12 from sklearn.isotonic import IsotonicRegression
13 from sklearn.utils import check_random_state
14
15 n = 100
16 x = np.arange(n)
17 rs = check_random_state(0)
18 y = rs.randint(-50, 50, size=(n,)) + 50. * np.log1p(np.arange(
19     (n)))
20 #
21 #####
22
23 # Fit IsotonicRegression and LinearRegression models
24
25 ir = IsotonicRegression()
26 y_ = ir.fit_transform(x, y)
27
28 lr = LinearRegression()
29 lr.fit(x[:, np.newaxis], y) # x needs to be 2d for
30 # LinearRegression
31 #####
32
33 # Plot result
34
35 segments = [[[i, y[i]], [i, y_[i]]] for i in range(n)]
36 lc = LineCollection(segments, zorder=0)
37 lc.set_array(np.ones(len(y)))
38 lc.set_linewidths(np.full(n, 0.5))
39
40 fig = plt.figure()

```

```

39 plt.plot(x, y, 'r.', markersize=12)
40 plt.plot(x, y_, 'b.-', markersize=12)
41 plt.plot(x, lr.predict(x[:, np.newaxis]), 'b-')
42 plt.gca().add_collection(lc)
43 plt.legend(('Data', 'Isotonic Fit', 'Linear Fit'), loc='lower
         right')
44 plt.title('Isotonic regression')
45 plt.show()

```

5. Kemudian jalankan aplikasi tersebut, dan bisa dicek hasil dari Variable explorernya

Name	Type	Size	Value
x	int	100	[1, ..., 100]
y	float64 (100,)	100	[0.0, ..., 0.9]
y_	float64 (100,)	100	[0.0, ..., 0.9]

Gambar 1.44 Variable Explorer

## 1.4.10 Mencoba Loading an example dataset

1. Sample kode

```

1 # -*- coding: utf-8 -*-
2 "" ""
3 Created on Wed Feb 26 19:32:36 2020
4
5 @author: lenovo
6 "" ""
7
8 from sklearn import datasets #import class/fungsi dataset
   dari scikit-learn library
9
10 import matplotlib.pyplot as plt #import matplotlib
11
12 #Load the digits dataset
13 digits = datasets.load_digits() #memuat dan memasukkan
   dataset digits ke variabel digits
14
15 #menampilkan digit pertama
16 plt.figure(1, figsize=(3, 3))
17 plt.imshow(digits.images[-1], cmap=plt.cm.gray_r,
   interpolation='nearest')
18 plt.show()

```

2. Kemudian jalankan aplikasi tersebut



Gambar 1.45 Dataset

### 1.4.11 Mencoba Learning and Predicting

#### 1. Sample kode

```

1  # -*- coding: utf-8 -*-
2  "" ""
3  Created on Thu Feb 27 19:07:08 2020
4
5  @author: lenovo
6  "" ""
7
8  from sklearn.linear_model import LogisticRegression #import
   dari library sklearn
9  from sklearn.datasets import make_blobs #import dari library
   sklearn
10
11 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
   random_state=1) # generate dataset klasifikasi 2d
12 # fit final model
13 model = LogisticRegression()
14 model.fit(X, y)
15
16 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
   random_state=1) # menentukan 1 buah contoh baru dimana
   jawabannya tidak diketahui
17
18 ynew = model.predict_proba(Xnew) # membuat prediksi
19 for i in range(len(Xnew)):
20     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #
   menampilkan hasil prediksi

```

#### 2. Kemudian jalankan aplikasi tersebut

```

In [18]: runfile('C:/Users/lenovo/Desktop/sample_1.py', wdir='C:/Users/lenovo/Desktop')
X: [-0.79451228  2.10095177], Predicted: [0.95467599 0.40512881]
X: [ 0.22980404  4.72855085], Predicted: [0.98221882 0.99751581]
X: [-2.18773168  3.13512125], Predicted: [0.99189073 0.80618927]

```

Gambar 1.46 Predicting

### 1.4.12 Mencoba Model Persistence

#### Sample kode

```

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

```

```

2 """
3 Created on Thu Feb 27 19:36:22 2020
4
5 @author: lenovo
6 """
7
8 from sklearn import svm #import dari library sklearn
9 from sklearn import datasets #import dari library sklearn
10 clf = svm.SVC() #menggunakan Support Vector Classifier
11 iris = datasets.load_iris() #menggunakan iris dataset
12 X, y = iris.data, iris.target
13 clf.fit(X, y)

```

### 1.4.13 Mencoba Conventions

Sample

kode

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 20:04:46 2020
4
5 @author: lenovo
6 """
7
8 import numpy as np #import dari library sklearn
9 from sklearn import random_projection #import dari library
   sklearn
10
11 rng = np.random.RandomState(0) #Menggunakan fungsi random
12 X = rng.rand(10, 2000) #mengambil nilai random
13 X = np.array(X, dtype='float32') #membuat array bertipe float32
14 X.dtype

```

## 1.5 1174050 Dika Sukma Pradana

### 1.5.1 Teori

1. Definisi, sejarah, dan perkembangan kecerdasan buatan.

Intelegensi buatan (AI) mengacu pada simulasi kecerdasan manusia dalam mesin yang diprogram untuk berpikir seperti manusia dan meniru tindakan mereka. Istilah ini juga dapat diterapkan pada mesin apa pun yang menunjukkan sifat-sifat yang terkait dengan pikiran manusia seperti pembelajaran dan pemecahan masalah.

Sejarah Kecerdasan Buatan (AI) bermula pada saat zaman kuno, dengan sejuta mitos, cerita dan desas-desus tentang makhluk buatan yang diberkahii dengan kecerdasan oleh pengrajin ahli. Benih-benih AI modern ditanam oleh para filsuf klasik yang mencoba menggam-

barkan proses pemikiran manusia sebagai manipulasi simbol secara mekanis. Karya ini memuncak dalam penemuan komputer digital yang dapat diprogram pada tahun 1940-an, sebuah mesin yang didasarkan pada esensi abstrak penalaran matematika. Perangkat ini dan ide-ide di belakangnya menginspirasi segelintir ilmuwan untuk mulai serius membahas kemungkinan membangun otak elektronik. Bidang penelitian AI didirikan pada lokakarya yang diadakan di kampus Dartmouth College selama musim panas 1956. Mereka yang hadir akan menjadi pemimpin penelitian AI selama beberapa dekade. Banyak dari mereka meramalkan bahwa sebuah mesin yang secerdas manusia akan hidup tidak lebih dari satu generasi dan mereka diberi jutaan dolar untuk mewujudkan visi atau tujuan ini. Akhirnya, menjadi jelas bahwa mereka meremehkan kesulitan proyek. Pada tahun 1973, sebagai tanggapan atas kritik dari James Lighthill dan tekanan terus-menerus dari kongres, AS dan Pemerintah Inggris menghentikan pendanaan penelitian yang tidak diarahkan pada kecerdasan buatan, dan tahun-tahun sulit berikutnya akan dikenal sebagai musim dingin AI. Tujuh tahun kemudian, sebuah inisiatif visioner oleh Pemerintah Jepang mengilhami pemerintah dan industri untuk menyediakan miliaran dolar dalam AI, tetapi pada akhir 80-an investor menjadi kecewa dengan kurangnya daya komputer (perangkat keras) yang dibutuhkan dan menarik lebih banyak dana. Investasi dan minat pada AI tumbuh pesat pada dekade pertama abad ke-21, ketika pembelajaran mesin berhasil diterapkan pada banyak masalah di dunia akademis dan industri karena metode baru, penerapan perangkat keras komputer yang kuat, dan kumpulan data yang sangat besar.

2. Definisi supervised learning, klasifikasi, regresi, dan unsupervised learning. Data set, training set dan testing set.

Supervised learning adalah sebuah metode pendekatan yang mana sudah terdapat data yang dilatih, dan terdapat variabel yang ditargetkan atau yang menjadi tujuan sehingga tujuan dari pendekatan ini adalah mengelompokkan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokkan data tersebut menjadi dua bagian atau bahkan tiga bagian dan seterusnya.

Klasifikasi adalah salah satu topik utama dalam data mining atau machine learning. Klasifikasi yaitu suatu pengelompokan data dimana data yang digunakan tersebut mempunyai kelas label atau target.

Regresi adalah Supervised learning tidak hanya mempelajari classifier, tetapi juga mempelajari fungsi yang dapat memprediksi suatu nilai numerik.

Data set merupakan sebuah cabang aplikasi dari Artificial Intelligence(AI)/Kecerdasan Buatan yang terfokus kepada pengembangan se-



buah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia(programmer).

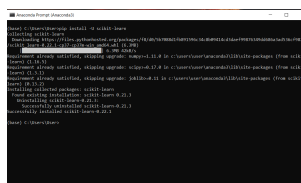
Training set yaitu jika pasangan objek, dan kelas yang menunjuk pada objek tersebut adalah suatu contoh yang telah diberi label akan menghasilkan suatu algoritma pembelajaran.

Testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar[?].

## 1.5.2 Instalasi

### 1.5.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu.
2. Install aplikasi Anaconda yang sudah di download tadi.
3. Simpan aplikasi sesuai folder yang kita pilih lalu next.
4. Centang Keduanya lalu tekan tombol install.
5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish.
6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall.
7. Kemudian ketikkan perintah `pip install -U scikit-learn` seperti gambar berikut.



Gambar 1.47 Instalasi

## 1.5.3 Percobaan

Mencoba Library

```

1 # -*- coding: utf-8 -*-
2 " " "
3 Created on Wed Feb 26 22:51:53 2020
4
5 @author: User
6 " " "
```

```

7
8 import matplotlib.pyplot as plt
9 from sklearn.svm import SVC
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.metrics import plot_roc_curve
12 from sklearn.datasets import load_wine
13 from sklearn.model_selection import train_test_split
14
15 X, y = load_wine(return_X_y=True)
16 y = y == 2
17
18 X_train, X_test, y_train, y_test = train_test_split(X, y,
19 random_state=42)
20 svc = SVC(random_state=42)
21 svc.fit(X_train, y_train)

```

SVC(random\_state=42)

**Gambar 1.48** Variabel Explore

```

1 from sklearn import datasets
2 #perintah untuk mengimport class datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #iris merupakan suatu estimator/parameter yang berfungsi untuk
   mengambil data pada item datasets.load iris
5 digits = datasets.load_digits()
6 #digits merupakan suatu estimator/parameter yang berfungsi untuk
   mengambil data pada item datasets.load digits
7 print(digits.data)
8 #perintah yang berfungsi untuk menampilkan estimator/parameter
   yang dipanggil pada item digits.data dan menampilkan
   outputannya
9 digits.target
10 #untuk mengambil target pada estimator/parameter digits dan
   menampilkan outputannya
11 digits.images[0]
12 #untuk mengambil images[0] pada estimator/parameter digits dan
   menampilkan outputannya

```

```

In [18]: runfile('C:/Users/User/Documents/Spyder/1.py', wdir='C:/Users/User/Documents/
Spyder')
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 15.  9.  0.]
 ...
 [ 0.  1. ...  5.  0.  0.]
 [ 0.  2. ... 12.  0.  0.]
 [ 0. 10. ... 12.  1.  0.]]

```

**Gambar 1.49** Datasets

```

1 # -*- coding: utf-8 -*-
2 "" ""
3 Created on Wed Feb 26 22:17:01 2020

```

```

4
5 @author: User
6 """
7 from sklearn import svm
8 #merupakan sebuah perintah untuk mengimport class svm dari
   packaged sklearn
9 from sklearn import datasets
10 # merupakan sebuah perintah untuk mengimport class datasets dari
   packaged sklearn
11 clf = svm.SVC(gamma='scale')
12 #clf sebagai estimator/parameter, svm.SVC sebagai class, gamma
   sebagai parameter untuk menetapkan nilai secara manual dengan
   nilai scale
13 iris = datasets.load_iris()
14 #iris sebagai estimator/parameter, datasets.load_iris() sebagai
   item dari suatu nilai
15 X, y = iris.data, iris.target
16 #X, y sebagai estimator/parameter, iris.data, iris.target sebagai
   item dari 2 nilai yang ada
17 clf.fit(X, y)
18 #clf sebagai estimator/parameter dengan menggunakan metode fit
   untuk memanggil estimator X, y dengan outputannya

```

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 23:06:05 2020
4
5 @author: User
6 """
7
8 import pickle
9 #pickle merupakan sebuah class yang di import
10 s = pickle.dumps(clf)
11 #s sebagai estimator/parameter dengan pickle.dumps merupakan
   suatu nilai/item dari estimator/parameter clf
12 clf2 = pickle.loads(s)
13 #clf2 sebagai estimator/parameter, pickle.loads sebagai suatu
   item, dan s sebagai estimator/parameter yang dipanggil
14 clf2.predict(X[0:1])
15 #clf2.predict sebagai suatu item dengan menggunakan metode
   predict untuk menentukan suatu nilai dari (X[0:1])
16 y[0]
17 #pada estimator/parameter y berapapun angka yang diganti nilainya
   akan selalu konstan yaitu 0
18
19
20 from joblib import dump, load
21 #sebuah perintah untuk mengimport class dump, load dari packaged
   joblib
22 dump(clf, 'filename.joblib')
23 #dump di sini sebagai class yang didalamnya terdapat nilai dari
   suatu item clf dan data joblib
24 clf = load('filename.joblib')
25 #clf sebagai estimato/parameter dengan suatu nilai load berfungsi
   untuk mengulang data sebelumnya

```

### 1.5.3.4 Conventions Type Casting

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Wed Feb 26 22:45:29 2020
4
5  @author: User
6  """
7
8  #Type Casting
9  from sklearn import svm
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
    svm dari packaged sklearn
11 from sklearn import random_projection
12 #pada baris ini merupakan sebuah perintah untuk mengimport class
    random_projection dari packaged sklearn
13 rng = np.random.RandomState(0)
14 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
    np.random.RandomState(0)
15 X = rng.rand(10, 2000)
16 #X sebagai estimator/parameter dengan nilai item rng.rand
17 X = np.array(X, dtype='float32')
18 #X sebagai estimator/parameter dengan nilai item np.array
19 X.dtype
20 #X.dtype sebagai item pemanggil
21 transformer = random_projection.GaussianRandomProjection()
22 #transformer sebagai estimator/parameter dengan memanggil class
    random_projection
23 X_new = transformer.fit_transform(X)
24 #X new di sini sebagai estimator/parameter dan menggunakan metode
    fit
25 X_new.dtype
26 #X new.dtype sebagai item
27
28
29 from sklearn import datasets
30 #pada baris ini merupakan sebuah perintah untuk mengimport class
    datasets dari packaged sklearn
31 from sklearn.svm import SVC
32 #pada baris ini merupakan sebuah perintah untuk mengimport class
    SVC dari packaged sklearn.svm
33 iris = datasets.load_iris()
34 #iris sebagai estimator/parameter dengan item datasets.load_iris
    ()
35 clf = SVC(gamma='scale')
36 #clf sebagai estimator/parameter dengan nilai class SVC pada
    parameter gamma sebagai set penilaian
37 clf.fit(iris.data, iris.target)
38 #estimator/parameter clf menggunakan metode fit dengan itemnya
39 list(clf.predict(iris.data[:3]))
40 #menambahkan item list dengan metode predict
41 clf.fit(iris.data, iris.target_names[iris.target])
42 #estimator/parameter clf menggunakan metode fit dengan itemnya
43 list(clf.predict(iris.data[:3]))
44 #menambahkan item list dengan metode predict

```

## Refitting and updating parameters

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Wed Feb 26 23:08:34 2020
4
5  @author: User
6  """
7
8  #Multiclass vs. Multilabel Fitting
9  from sklearn.svm import SVC
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
    SVC dari packaged sklearn.svm
11 from sklearn.multiclass import OneVsRestClassifier
12 #pada baris ini merupakan sebuah perintah untuk mengimport class
    OneVsRestClassifier dari packaged sklearn.multiclass
13 from sklearn.preprocessing import LabelBinarizer
14 #pada baris ini merupakan sebuah perintah untuk mengimport class
    LabelBinarizer dari packaged sklearn.preprocessing
15 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
16 y = [0, 0, 1, 1, 2]
17 classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
    random_state=0))
18 classif.fit(X, y).predict(X)
19 y = LabelBinarizer().fit_transform(y)
20 classif.fit(X, y).predict(X)
21
22
23 from sklearn.preprocessing import MultiLabelBinarizer
24 y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
25 y = MultiLabelBinarizer().fit_transform(y)
26 classif.fit(X, y).predict(X)

```

## Multiclass vs. multilabel fitting

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Wed Feb 26 23:10:03 2020
4
5  @author: User
6  """
7
8  #Refitting and Updating Parameters
9  import numpy as np
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
    svm dari np
11 from sklearn.svm import SVC
12 #pada baris ini merupakan sebuah perintah untuk mengimport class
    SVC dari packaged sklearn.svm
13 rng = np.random.RandomState(0)
14 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
    np.random.RandomState(0)
15 X = rng.rand(100, 10)
16 #X sebagai estimator/parameter dengan nilai item rng.rand
17 y = rng.binomial(1, 0.5, 100)
18 #y sebagai estimator/parameter dengan nilai item rng.binomial
19 X_test = rng.rand(5, 10)
20 #X test sebagai estimator/parameter dengan nilai item rng.rand

```

```

21 clf = SVC()
22 #clf sebagai estimator/parameter dan class SVC
23 clf.set_params(kernel='linear').fit(X, y)
24 #set params sebagai item
25 clf.predict(X_test)
26 #menggunakan metode predict
27 clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
28 clf.predict(X_test)

```

## 1.5.4 Penanganan error

```

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 "C:\Users\User\Documents\Spyder/2.py", line 14
    X, y = iris.data, iris.target
    ^
IndentationError: unexpected indent

```

**Gambar 1.50** Error

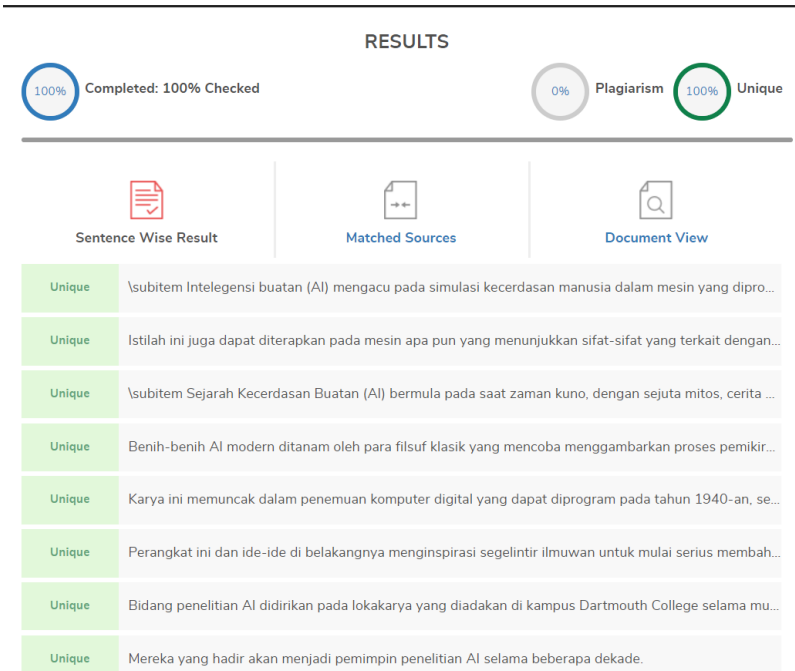
### 1.5.4.1 ScreenShoot Error

### 1.5.4.2 Tuliskan Kode Error dan Jenis Erornya

IndentationError: unexpected indent

**1.5.4.3 Solusi Pemecahan Masalah Error** Pastikan semua spasi pada koding sama. Menggunakan spasi atau tab.

## 1.5.5 Plagiarism



Gambar 1.51 Plagiarism

## 1.6 1174057 Alit Fajar Kurniawan

### 1.6.1 Teori

**1.6.1.1 Sejarah Perkembangan dan Definisi Artificial Intelligence** Kecerdasan buatan merupakan sebuah bidang dalam ilmu computer yang begitu penting di zaman ini dan masa yang akan datang guna mewujudkan sebuah sistem computer yang begitu cerdas. Artificial Intelligence atau biasa di singkat dengan AI berasal dari bahasa latin yang dimana intelligence berarti saya paham.

Pada tahun 1955, Newell dan juga Simon telah mengembangkan The Logic Theorist, yaitu program AI pertama. Dimana program tersebut mempresen-tasikan sebuah masalah sebagai model pohon, lalu diselesaikan dengan cara memilih cabang yang akan mewujudkan kesimpulan terbenar dan tepat. Program AI tersebut berdampak sangat besar dan dapat mendaji batu loncatan yang cukup penting dalam mengembangkan bidang AI [?].

Masa Perkembangan AI dimulai pada awal era komputer elektronik pada tahun 1941. dimana ditemukannya alat penyimpanan dan pemrosesan in-formasi. kemudian dilanjutkan pada masa-masa persiapan AI yang terjadi

pada tahun 1943-1956. Pada sekitaran tahun 1952-1969 merupakan masa awal perkembangan AI terjadi, dan pada tahun 1966-1974 perkembangan AI mengalami penurunan atau melambatnya proses dalam melakukan pengembangan. pada tahun 1969 sampai 10 tahun kedepan kembali terjadi perkembangan yang menciptakan inovasi sistem berbasis pengetahuan. dan sekitaran tahun 1980-an AI kembali menjadi sebuah industri yang terus berkembang sampai sekarang ini.

### *1.6.1.2 learning, klasikasi, regresi dan unsupervised learning. Data set, training set dan testing set*

#### 1. Definisi Supervised Learning Dan Unsupervised Learning

Supervised Learning merupakan suatu pendekatan yang dimana terdapat data dan variable yang telah ditargetkan sehingga pendekatan tersebut bertujuan untuk dapat mengelompokkan sebuah data ke data yang sudah ada, beda dengan Unsupervised learning yang tidak mempunyai data, sehingga data yang ada harus di kelompokkan menjadi beberapa bagian.

#### 2. Definisi Klasifikasi Dan Regresi

Klasifikasi adalah sebuah kegiatan penggolongan atau pengelompokan. Menurut kamus besar bahasa Indonesia yang dimana klasifikasi merupakan penyusunan sistem di dalam kelompok atau golongan berdasarkan kaidah atau standar yang telah ditetapkan. Regresi adalah sebuah metode analisis statistic yang akan digunakan untuk melihat pengaruh variable.

#### 3. Devinisi Dataset, Training Set, Dan Testing Set

Dataset adalah sebuah objek yang akan mempresentasikan sebuah data dan relasinya di memory. Struktur pada dataset ini mirip dengan data yang ada di dalam database. Training set adalah bagian dari dataset yang berperan dalam membuat prediksi atau algoritma sesuai tujuan masing-masing. Testing set adalah bagian dari dataset yang akan di tes guna melihat keakuratan atau ketepatan datanya.

## **1.6.2 Praktek**

### *1.6.2.1 Instalasi*

1. Melakukan instalasi library scikit pada anaconda, ketik kan pip install -U scikit-learn pada terminal anaconda.



```

Anaconda Prompt (Anaconda3)

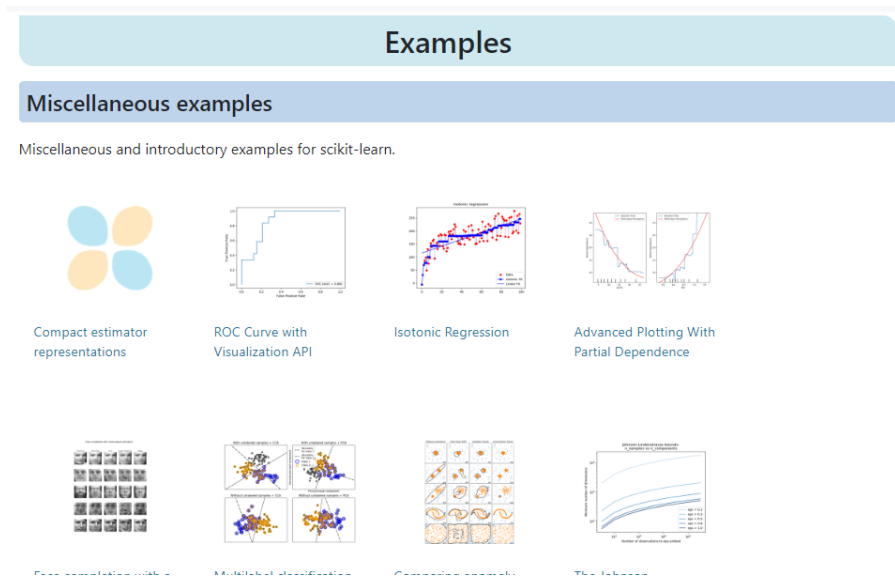
(base) C:\Users\alitif>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/f8/d0/5b7088d1fb891596c34c8b09414cd3daef99876349dd686a3ad536cf9820/scikit_learn-0.22.1-cp37-cp37m-win_amd64.whl (6.3MB)
    |#####| 6.3MB 297kB/s
Requirement already satisfied, skipping upgrade: numpy>=1.11.0 in c:\users\alitif\anaconda3\lib\site-packages (from scikit-learn) (1.16.4)
Requirement already satisfied, skipping upgrade: joblib>=0.11 in c:\users\alitif\anaconda3\lib\site-packages (from scikit-learn) (0.13.2)
Requirement already satisfied, skipping upgrade: scipy>=0.17.0 in c:\users\alitif\anaconda3\lib\site-packages (from scikit-learn) (1.2.1)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.21.2
  Uninstalling scikit-learn-0.21.2:
    Successfully uninstalled scikit-learn-0.21.2
Successfully installed scikit-learn-0.22.1

(base) C:\Users\alitif>

```

Gambar 1.52 Instalasi Scikit Learn

- Setelah selesai instalasi, pilih salah satu example dari website Scikit.



Gambar 1.53 Example

```

1 print(__doc__)
2
3 from sklearn.linear_model import LogisticRegression
4 from sklearn import set_config
5
6
7 lr = LogisticRegression(penalty='l1')
8 print('Default representation:')

```

```

9 print(lr)
10 # LogisticRegression(C=1.0, class_weight=None, dual=False,
    fit_intercept=True,
11 #                      intercept_scaling=1, l1_ratio=None,
    max_iter=100,
12 #                      multi_class='auto', n_jobs=None, penalty
    ='l1',
13 #                      random_state=None, solver='warn', tol
    =0.0001, verbose=0,
14 #                      warm_start=False)
15
16 set_config(print_changed_only=True)
17 print('\nWith changed_only option:')
18 print(lr)
19 # LogisticRegression(penalty='l1')

```

kemudian coba jalankan, lihat hasilnya

@author: alitf

Default representation:

LogisticRegression(penalty='l1')

With changed\_only option:

LogisticRegression(penalty='l1')

Gambar 1.54 Example

### 3. latihan 2 Mencoba Loading an example dataset

```

1 from sklearn import datasets #mengimport class dataset dari
    scikit learn library
2 iris = datasets.load_iris() # memuat dan memasukkan dataset
    iris ke variabel bernama iris
3 digits = datasets.load_digits() #memuat dan memasukkan
    dataset digits ke variabel digits
4
5 print(digits.data) #memberikan akses ke fitur yang dapat
    digunakan untuk mengklasifikasikan sampel digit dan
    menampilkannya di console
6
7 digits.target #memberikan informasi tentang data yang
    berhubungan atau juga dapat dijadikan sebagai label
8
9 digits.images[0] #Data selalu berupa array 2D, shape (
    n_samples, n_features), meskipun data aslinya mungkin
    memiliki bentuk yang berbeda.

```

hasil dari data digits

```
In [17]: runfile('C:/Users/alitf/OneDrive/Desktop/AI/src/1174057
wdir='C:/Users/alitf/OneDrive/Desktop/AI/src/1174057/chapter1')
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10. 0.  0.]
 [ 0.  0.  0. ... 16. 9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12. 0.  0.]
 [ 0.  0. 10. ... 12. 1.  0.]]
```

Gambar 1.55 Result Data Digits

hasil dari digits.target

```
In [19]: digits.target #memberikan informasi:
          dijadikan sebagai label
Out[19]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.56 Result digits.target

hasil dari digits.image

```
In [20]: digits.images[0] #Data selalu berupa array 2D, s
          meskipun data aslinya mungkin memiliki bentuk yang berbec
Out[20]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

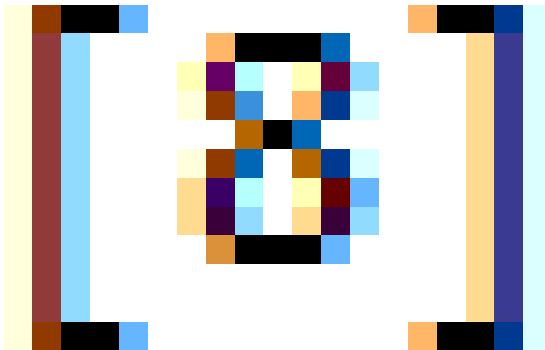
Gambar 1.57 Result digits.image

```

1 from sklearn import svm # perintah untuk mengimport class svm
  dari packaged sklearn
2
3 digits = datasets.load_digits() #memuat dan memasukkan
  dataset digits ke variabel digits
4
5 clf = svm.SVC(gamma=0.001, C=100.) #clf sebagai estimator/
  parameter , svm.SVC sebagai class , gamma sebagai
  parameter untuk menetapkan nilai secara manual
6
7 clf.fit(digits.data[:-1], digits.target[:-1]) # clf sebagai
  estimator/parameter , f i t sebagai metode , digits . data
  sebagai item , [:1] sebagai syntax pythonnya dan
  menampilkan outputannya
8
9 print(clf.predict(digits.data[-1:])) #clf sebagai estimator/
  parameter , predict sebagai metode lainnya , digits . data
  sebagai item dan menampilkan outputannya

```

kemudian coba jalankan, lihat hasilnya



**Gambar 1.58** Result Learning and predicting

#### 5. latihan 4 Mencoba Model persistence

```

1 from sklearn import svm, datasets #mengimport class dataset
  dari scikit learn library

```

```

2 clf = svm.SVC(gamma=0.001, C=100.) #memanggil class SVC dan
   menset argument constructor SVC serta ditampung di
   variable clf
3 X, y = datasets.load_iris(return_X_y=True) #meload datasets
   iris dan ditampung di variable x untuk data dan y untuk
   target
4 clf.fit(X, y) #memanggil method fit untuk melakukan training
   data dengan argumen data dan target dari datasets iris
5
6 #Pickle
7 import pickle #mengimport pickle
8 s = pickle.dumps(clf) #memanggil method dumps dengan argumen
   clf dan ditampung di variable s
9 clf2 = pickle.loads(s) #memanggil method loads dengan argumen
   s dan ditampung di variable clf2
10 print(clf2.predict(X[0:1])) #menampilkan hasil dari method
   predict dengan argumen data variable X pertama
11
12 #Joblib
13 from joblib import dump, load #mengimport dump dan load dari
   library joblib
14 dump(clf, '1174057.joblib') #memanggil method dumps dengan
   argumen clf dan nama file joblibnya
15 clf3 = load('1174057.joblib') #memanggil method loads dengan
   argumen nama file joblibnya dan ditampung di variable clf3
16 print(clf3.predict(X[0:1])) #menampilkan hasil dari method
   predict dengan argumen data variable X pertama

```

kemudian coba jalankan, lihat hasilnya

```

In [13]: runfile('D:/Data Alit/Kuliah,
modelpersistence.py', wdir='D:/Data A
[0]
[0]

```

**Gambar 1.59** Result Model persistence

## 6. latihan 5 Mencoba Conventions

```

1 from sklearn import svm, datasets #mengimport class dataset
   dari scikit learn library
2 clf = svm.SVC(gamma=0.001, C=100.) #memanggil class SVC dan
   menset argument constructor SVC serta ditampung di
   variable clf
3 X, y = datasets.load_iris(return_X_y=True) #meload datasets
   iris dan ditampung di variable x untuk data dan y untuk
   target
4 clf.fit(X, y) #memanggil method fit untuk melakukan training
   data dengan argumen data dan target dari datasets iris

```

```

5
6 #Pickle
7 import pickle #mengimport pickle
8 s = pickle.dumps(clf) #memanggil method dumps dengan argumen
   clf dan ditampung di variable s
9 clf2 = pickle.loads(s) #memanggil method loads dengan argumen
   s dan ditampung di variable clf2
10 print(clf2.predict(X[0:1])) #menampilkan hasil dari method
   predict dengan argumen data variable X pertama
11
12 #Joblib
13 from joblib import dump, load #mengimport dump dan load dari
   library joblib
14 dump(clf, '1174057.joblib') #memanggil method dumps dengan
   argumen clf dan nama file joblibnya
15 clf3 = load('1174057.joblib') #memanggil method loads dengan
   argumen nama file joblibnya dan ditampung di variable clf3
16 print(clf3.predict(X[0:1])) #menampilkan hasil dari method
   predict dengan argumen data variable X pertama

```

kemudian coba jalankan, lihat hasilnya

```

In [20]: runfile('C:/Users/Alit/Desktop/KB3A/src/1174006/chapter1/coba4.py', wdir='C:/
Users/Alit/Desktop/KB3A/src/1174006/chapter1')
float32
float64
[0, 0, 0]
['setosa', 'setosa', 'setosa']
[0 0 0 0 0]
[0 0 0 0 0]
[0 0 1 1 2]
[[1 0 0]
 [1 0 0]
 [0 1 0]
 [0 0 0]
 [0 0 0]]
[[1 0 1 0 0]
 [1 0 1 0 0]
 [1 0 1 1 0]
 [1 0 1 0 0]
 [1 0 1 0 0]]

```

Gambar 1.60 Result Conventions

### 1.6.3 Penanganan Error

#### 1. Screenshot Error

```

File "D:/Data Alit/Kuliah/SEMESTER VI/AI/src/1174057/chapter1/learning.py", line 5, in
<module>
    clf.fit(digits.data[:-1], digits.target[:-1]) # clf sebagai estimator/parameter , f
i t sebagai metode , digits . data sebagai item , [:1] sebagai syntax pythonnya dan
menampilkan outputannya

NameError: name 'digits' is not defined

```

Gambar 1.61 Error

```
File "D:/Data Alit/Kuliah/SEMESTER VI/AI/src/1174057/chapter1/learning.py", line 3
    digits = datasets . load digits () #meloadd datasets digits dan ditampung di variable
    digits
SyntaxError: invalid syntax
```

Gambar 1.62 Error

## 2. Tuliskan kode dan jenis error

is not defined, xception yang terjadi saat syntax melakukan eksekusi terhadap local name atau global name yang tidak terdenisi.

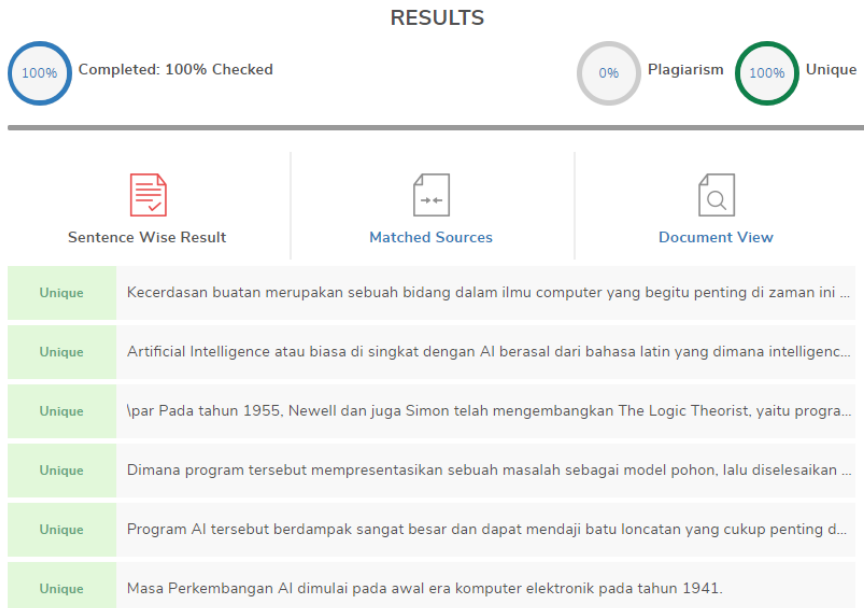
invalid syntax

## 3. Solusi penanganan error

Solusinya adalah memastikan variabel atau function yang dipanggil ada atau tidak salah ketik.

Periksa kembali syntax yang dibuat, bisa saja ada kesalahan dalam spasi.

### 1.6.4 Bukti Tidak Plagiat



Gambar 1.63 Plagiarisme

## 1.7 1174039 - Liyana Majdah Rahma

### 1.7.1 Teori

**1.7.1.1 Definisi Kecerdasan Buatan** Artificial Intelligence adalah kecerdasan yang ditambahkan pada suatu sistem yang dapat diatur dalam konteks secara ilmiah. Menurut Michael Haelein menyatakan bahwa AI merupakan “sistem yang mempunyai kemampuan untuk menguraikan, belajar agar dapat digunakan untuk mencapai tujuan dengan melalui adaptasi yang fleksibel”. Kecerdasan buatan juga dapat dibuat dan dimasukkan ke dalam mesin agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia dengan cepat dan tepat.

**1.7.1.2 Sejarah Kecerdasan Buatan** Sejarah Kecerdasan buatan pertama kali terjadi pada zaman kuno, terdapat mitos ataupun cerita dan desas-desus tentang sebuah makhluk yang mempunyai kecerdasan serta kesadaran yang diberikan oleh pengrajin. Benih - benih nya mulai ditanam oleh para filsuf klasik yang mencari cara untuk menggambarkan proses berfikir manusia sebagai manipulasi simbol secara mekanis yang memuncak pada penemuan komputer digital di tahun 1940-an, yaitu sebuah mesin yang didasarkan penalaran matematika. Istilah kecerdasan buatan sendiri baru muncul pada tahun 1956, dan teori -teori nya sudah muncul sejak tahun 1941. Dan Pada tahun 1973, sebagai tanggapan atas kritik dari James Lighthill dan tekanan terus-menerus dari kongres, AS dan Pemerintah Inggris menghentikan pendanaan penelitian yang tidak diarahkan pada kecerdasan buatan, dan tahun-tahun sulit berikutnya akan dikenal sebagai musim dingin AI. Tujuh tahun kemudian, sebuah inisiatif visioner oleh Pemerintah Jepang mengilhami pemerintah dan industri untuk menyediakan miliaran dolar dalam AI, tetapi pada akhir 80-an investor menjadi kecewa dengan kurangnya daya komputer (perangkat keras) yang dibutuhkan dan menarik lebih banyak dana.

### 1.7.1.3 Perkembangan Kecerdasan Buatan

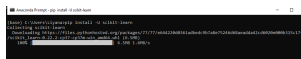
1. Pada tahun 1958, McCarthy di MIT AI Lab Memo orang pertama yang mendefinisikan bahasa pemrograman pada tingkat tinggi yaitu LISP, yang sekarang mendominasi pembuatan program-program kecerdasan buatan.
2. Kemudian Pada tahun 1959, Nathaniel Rochester dari IBM serta mahasiswa-mahasiswanya mengeluarkan program kecerdasan buatan yaitu Geometry Theorm Prover. selain itu juga Program ini dapat mengeluarkan suatu teorema menggunakan aksioma-aksioma yang ada.
3. Sistem Berbasis Pengetahuan pada tahun 1969-1979
4. Kecerdasan Buatan menjadi sebuah industri pada tahun 1980-1988. Kembalinya Jaringan Syaraf tiruan pada tahun 1986-sekarang.



## 1.7.2 Instalasi

### 1.7.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu.
2. Install aplikasi Anaconda yang sudah di download tadi.
3. Simpan aplikasi sesuai folder yang kita pilih lalu next.
4. Centang Keduanya lalu tekan tombol install.
5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish.
6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall.
7. Kemudian ketikkan perintah `pip install -U scikit-learn` seperti gambar berikut.



**Gambar 1.64** Instalasi

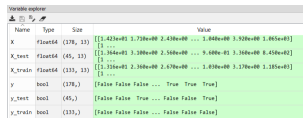
Pilih salah satu example dari website tersebut lalu jalankan

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun Mar  1 23:18:58 2020
4
5  @author: Liyana
6  """
7
8  import matplotlib.pyplot as plt
9  from sklearn.svm import SVC
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.metrics import plot_roc_curve
12 from sklearn.datasets import load_wine
13 from sklearn.model_selection import train_test_split
14
15 X, y = load_wine(return_X_y=True)
16 y = y == 2
17
18 X_train, X_test, y_train, y_test = train_test_split(X, y,
19                                                    random_state=42)
20 svc = SVC(random_state=42)
21 svc.fit(X_train, y_train)

```

buka variable `explolernya`



Name	Type	Size	Value
x_train	float64	(178, 21)	[[ 1.4126e+01 1.7184e+00 2.4484e+00 ... 1.8480e+00 1.3264e+00 1.8850e+01]]
x_test	float64	(45, 21)	[ 1 ...]
y_train	float64	(178, 1)	[[ 1.1644e+01 1.3880e+00 2.5480e+00 ... 1.4080e+01 1.3264e+00 1.4164e+01]]
y_test	float64	(45, 1)	[ 1 ...]
x_train	float64	(178, 21)	[[ 1.1164e+01 2.3880e+00 2.4780e+00 ... 1.4180e+00 1.3764e+00 1.1164e+01]]
y_train	float64	(178, 1)	[ 1 ...]
y	bool	(178,)	[False False False ... True True True]
x_test	bool	(45,)	[False False True ... False False True]
y_test	bool	(45,)	[False False True ... False False True]
x_train	bool	(178,)	[False False False ... False False False]

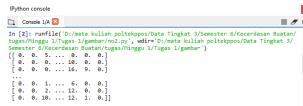
Gambar 1.65 Variable Exploler

### 1.7.2.2 Mencoba loading an example dataset

1. mengambil data iris dan digit dari dataset

```
1 iris = datasets.load_iris() # Dapat membuat sebuah variable
   iris yang mempunyai isi yaitu dataset iris
2 digits = datasets.load_digits() # Digunakan untuk membuat
   sebuah variable digits yang mempunyai isi yaitu dataset
   digits
```

2. Menampilkan data digits



```

In [2]: from sklearn.datasets import load_digits
Out[2]: (array([[0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               ...,
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.],
               [0., 0., 0., ..., 0., 0.]]),
         array([0, 1, 2, ..., 8, 9, 8]))

```

Gambar 1.66 Data Digits

3. menampilkan digits.target

```
In [16]: digits.target
Out[16]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.67 Digits Target

4. menampilkan data bentuk 2D.

```
In [17]: digits.images[0]
Out[17]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

Gambar 1.68 Data 2D

### 1.7.2.3 Mencoba Learning and Predicting

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun Mar  1 23:29:41 2020
4
5  @author: Liyana
6  """
7
8  from sklearn.linear_model import LogisticRegression # digunakan
   untuk mengimport linear_model dari library sklearn
9  from sklearn.datasets import make_blobs #digunakan untuk
   mengimport library datasets dari sklearn
10
11 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
   random_state=1) # dapat untuk generate dataset dengan
   klasifikasi 2D
12 model = LogisticRegression() # dapat menggunakan metode logistic
   regression
13 model.fit(X, y)
14
15 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
   random_state=1) # dapat menentukan 1 buah contoh baru dimana
   jawabannya tidak dapat diketahui
16
17 ynew = model.predict_proba(Xnew) # membuat sebuah prediksi dan
   memasukkannya kedalam variable ynew
18 for i in range(len(Xnew)):
19     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #menampilkan
   hasil prediksi

```

### 1.7.2.4 Mencoba Model Persistence

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun Mar  1 23:30:06 2020
4
5  @author: Liyana
6  """
7
8  from sklearn import svm #dapat mengimport svm dari library
   sklearn
9  from sklearn import datasets #digunakan untuk mengimport datasets
   dari library sklearn
10 clf = svm.SVC() # digunakan dengan menggunakan method SVC
11 iris = datasets.load_iris() # digunakan dengan menggunakan
   dataset iris
12 X, y = iris.data, iris.target #memasukkan x sebagai iris data ,
   dan y sebagai iris target
13 clf.fit(X, y) #laalu menggunakan metod fit .

```

### 1.7.2.5 Mencoba Conventions

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sun Mar  1 23:31:24 2020
4
5 @author: Liyana
6 """
7
8 import numpy as npy #mengimport numpy sebagai npy
9 from sklearn import random_projection #mengimport
   random_projection dari library sklearn
10
11 rng = npy.random.RandomState(0) #Menggunakan fungsi random dari
   numpy
12 X = rng.rand(10, 2000) #membuat range random diantara 10 sampai
   2000
13 X = npy.array(X, dtype='float32') #yang dijadikan array dengan
   tipe data float32
14 X.dtype

```

### 1.7.3 Penanganan Error

ValueError: found array with 0 sample(s) (shape=(0, 2)) while a minimum of 1 is required.

**Gambar 1.69** Data 2D

#### 1.7.3.1 Screenshot Error

#### 1.7.3.2 Kode error dan jenis error Jenis errornya adalah value error

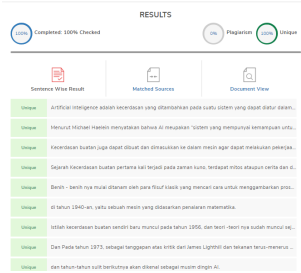
```

1 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
   random_state=1) # dapat menentukan 1 buah contoh baru dimana
   jawabannya tidak dapat diketahui

```

#### 1.7.3.3 Solusi Error Solusinya adalah dengan menggantikan nilai nya adalah n\_samples nya agar tidak 0

1.7.4 Cek Plagiarism



Gambar 1.70 Cek Plagiarism

## BAB 2

---

## CHAPTER 2

---

### 2.1 1174042 Faisal Najib Abdullah

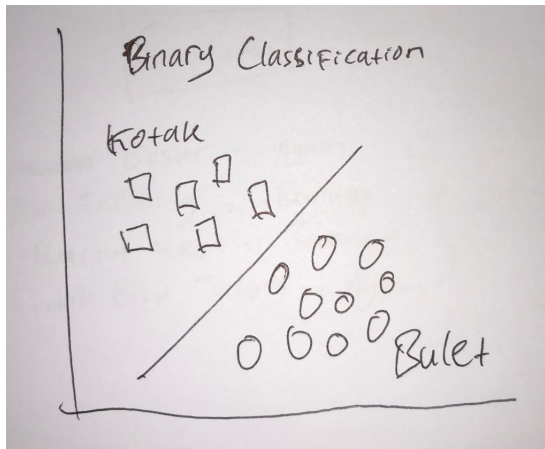
#### 2.1.1 Teori

1. Jelaskan Apa Itu binari classification dilengkapi ilustrasi gambar sendiri.

Binary Classification atau binomial adalah tugas mengklasifikasikan unsur-unsur dari himpunan yang diberikan ke dalam kedua kelompok berdasarkan aturan klasifikasi yang telah ditetapkan. Binari classification juga dapat diartikan sebagai pembagi yang hanya memberikan dua pilihan contohnya benar dan salah atau klasifikasi tingkat panjang atau pendek. Penjelasan lebih singkatnya binari classification merupakan kegiatan mengklasifikasikan yang hanya memberikan dua class.

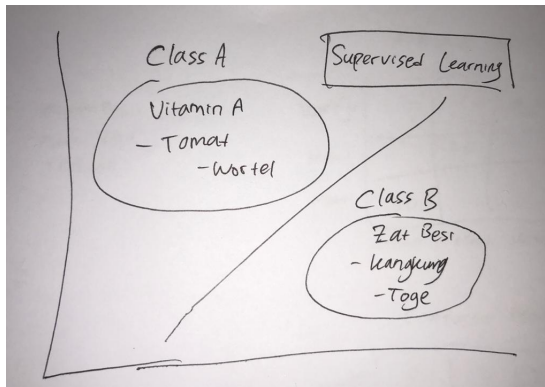
2. Jelaskan Apaitu supervised learning, unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

Supervised learning adalah cara untuk mengklasifikasikan suatu objek atau data yang telah ditentukan kelas-kelasnya. Contoh pada sayuran, tumbuhan wortel termasuk yang mengandung vitamin A berarti tum-



**Gambar 2.1** contoh binari calssification

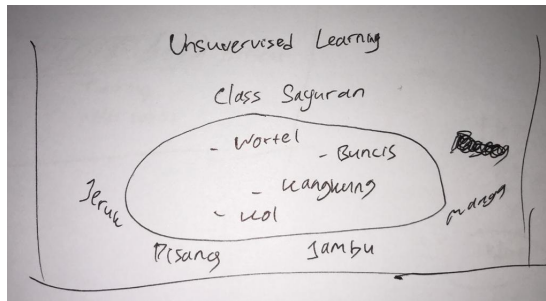
buhan wortel telah di kategorikan kedalam sayuran yang mengandung vitamin A. sedangkan kangkung mengandung zat besi yang berarti tumbuhan kangkung telah di kategorikan kedalam sayuran yang mengandung zat besi untuk lebih jelasnya dapat dilihat pada gambar.



**Gambar 2.2** contoh supervised learning

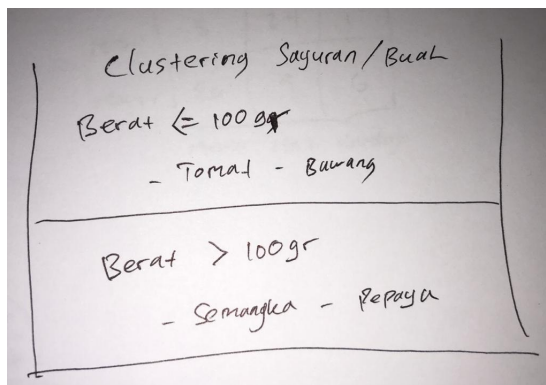
unsupervised learning merupakan cara untuk mengklasifikasi tanpa adanya kelas untuk menentukan jenisnya contoh sayuran berarti semua objek yang memiliki ciri ciri sayuran di kategorikan kedalam sayuran untuk lebih jelasnya dapat dilihat pada gambar.

clustering merupakan peroses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuannya contoh pada berat sayuran sayuran A memiliki berat 100 gr dan sayuran B memiliki berat 120 gr yang berarti



**Gambar 2.3** contoh unsupervised learning

berat sayuran dibagi dua parameter yaitu lebih kecil samadengan 100 gram dan lebih besar dari gram contoh pada gambar.

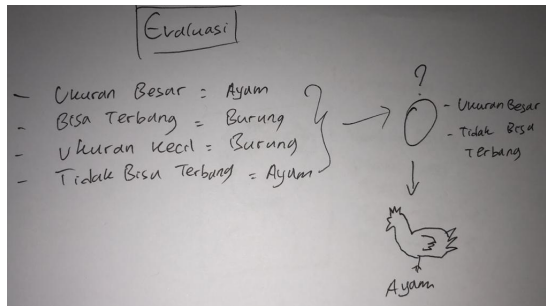


**Gambar 2.4** contoh clustering

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

evaluasi adalah pengumpulan dan pengamatan dari berbagai macam bukti untuk mengukur dampak efektifitas dari suatu objek, program, atau proses berkaitan dengan spesifikasi atau persyaratan yang telah di tetapkan sebelumnya. sedangkan akurasi itu sendiri merupakan bagian dari evaluasi yang merupakan ketepatan data terhadap suatu objek berdasarkan kriteria tertentu. kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. ketepatan akan di definisikan sebagai presentase kasus yang di klasifikasikan dengan benar. hal ini berkaitan dengan confusion matrix pada materi selanjutnya. contoh evaluasi untuk membedakan burung dengan ayam terdapat parameter yaitu ukuran badan dan fungsi sayap pada hewan tersebut. lebih jelasnya pada gambar berikut:





**Gambar 2.5** contoh evaluasi dan akurasi

4. Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Dalam pembuatan confusion matrix tentukan parameter atau objek yang akan di evaluasi contoh bunga melati , bunga mawar, dan bunga kenangan buat tabel dengan baris dan kolom berjumlah tiga kemudian tentukan nilai miring pada setiap kolom tersebut disini saya memberi nilai 30 dengan ketentuan setiap baris harus berisi nilai 30 nilai tersebut jika terbagi ke kolom lain maka jumlahnya harus bernilai 30 jika tidak berarti data tersebut tidak akurat. untuk lebih jelasnya dapat dilihat pada gambar berikut :

Confusion Matrix

Kamboja			30
Melati		30	
Mawar	30		
	Mawar	Melati	Kamboja

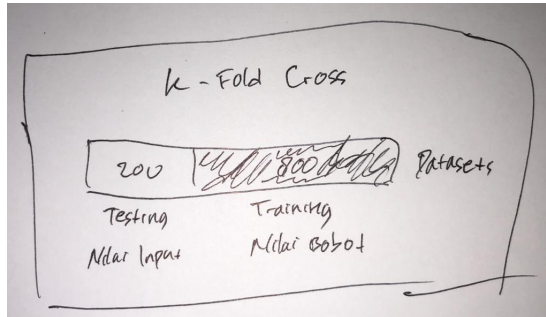
  

Confusion Matrix

Kamboja	5	2	23
Melati	5	24	1
Mawar	20	4	6
	Mawar	Melati	Kamboja

**Gambar 2.6** contoh Confusion Matrix

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 200 data digunakan untuk data testing kemudian 800 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukkan kedalam rumus regresi linier. sedangkan nilai testing akan dijadikan nilai inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar berikut :



**Gambar 2.7** contoh K-fold cross validation

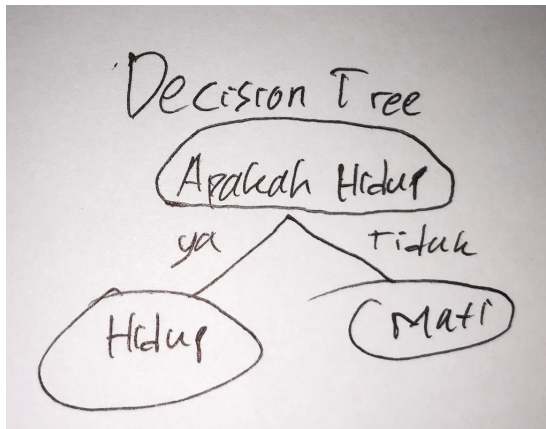
6. Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

Decision tree (pohon keputusan) merupakan implementasi dari binari classification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh pada root berisi nilai jenis kelamin, apakah perempuan pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti jenis kelaminnya perempuan dan jika tidak maka bernilai laki-laki. agar lebih jelas dapat dilihat pada gambar decision tree berikut:

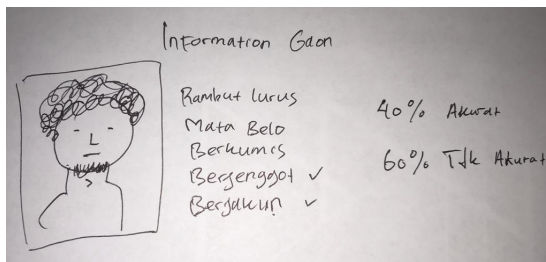
7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

informasion gain merupakan informasi atau kriteria dalam pembagian sebuah objek contoh information gain pada laki-laki yaitu berrambut lurus, mata belo, berkumis, berjenggot, dan memiliki jakun. untuk lebih jelasnya dapat dilihat pada gambar berikut :

sedangkan entropi merupakan ukuran keacakan dari informasi semakin tinggi entropi maka semakin sulit dalam menentukan keputusan contoh dalam menentukan jenis kelamin semakin detail informasi maka akan semakin susah dalam menentukan keputusan.



**Gambar 2.8** contoh decision tree



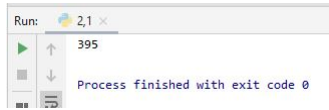
**Gambar 2.9** contoh information gain

### 2.1.2 Sikic-Learn

1. pada surcode pertama yang dapat dilihat pada gambar pada baris pertama di tuliskan yang berarti mengimport library padas. selanjutnya pada baris ke dua codingan tersebut berisi pada code tersebut terdapat variabel muaraenim yang berisi inialisasi padas dengan aksi untuk membaca vfile csv yang terdapat pada direktori pada komputer kemudian terdapat sep samadengan titik koma yang berarti pemisah field di dalam vile tersebut merupakan titik koma. kemudian pada bagian akhir terdapat code len (nama variabel) yang berarti akan menghotung jumlah baris pada file tersebut.

```

1 # load dataset (student mat pakenya)
2 import pandas as pd
3 muaraenim = pd.read_csv('D://NAJIB/SEMESTER_6_NAJIB/AI/
    Chapter2/dataset/student-mat.csv', sep=';')
4 print(len(muaraenim))
  
```



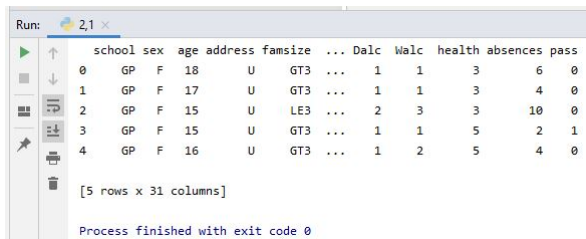
Gambar 2.10 hasil

2. Pada code selanjutnya akan ditambahkan fungsi untuk lulus atau gagal yang dibuat berupa kolom kolom yang di dalamnya terdapat nilai 0 dan 1 dimana 0 berarti gagal dan 1 berarti lulus. kemudian variabel pada codingan sebelumnya masih digunakan. dimana variabel muaraenim digunakan karena berisi nilai file csv kemudian dilakukan eksekusi dengan parameter G1, G2, dan G3 dengan fungsi lambda yang berarti if bersarang atau if didalam if yang berarti nilai yang di eksekusi akan dilempar ke dalam setiap paramater sasuai dengan kriteria dan axis=1 yaitu nilai tersebut akan digunakan dari tiap baris data. sedangkan pada codingan selanjutnya variabel muaraenim di berikan aksi drop yaitu penurunan nilai pada bagian baris data. dan pada code muaraenim.head () yaitu untuk mengeksekusi codingan sebelumnya.

```

1 # generate binary label (pass/fail) based on G1+G2+G3
2 # (test grades, each 0-20 pts); threshold for passing is sum
  >=30
3 muaraenim['pass'] = muaraenim.apply(lambda row: 1 if (row['G1
  ']+row['G2']+row['G3'])
4 >= 35 else 0, axis=1)
5 muaraenim = muaraenim.drop(['G1', 'G2', 'G3'], axis=1)
6 print(muaraenim.head())

```



Gambar 2.11 hasil

3. pada kodingan selanjutnya digunakan untuk menambahkan nilai numerik berupa 0 dan 1 yang dirubah dari nilai tidak dan iya hal ini merupakan fungsi dari codingan get\_dummies pada baris pertama yang nilainya diambil dari variabel muaraenim yang telah di dekralisasi tadi banyaknya data numerik itu sendiri tergantung pada field dari kolom yang di cam-tumkan pada codingan dengan catatan field tersebut harus ada dalam data csv yang di import oleh codingan pertama tadi maka hasilnya akan merubah data dalam field tersebut menjadi 0 dan 1.

```

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

```

	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 59 columns]

Process finished with exit code 0

**Gambar 2.12** hasil

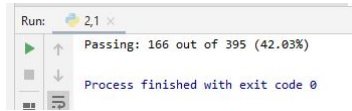
- selanjutnya pada codingan selanjutnya yaitu menentukan data training dan data testing dari dataset dimana variabel muaraenim yang berisi data csv tadi dibuat perbandingan 500 untuk data training dan sisanya yaitu 149 digunakan untuk data testing hal ini di lakukan pada baris ke 1 sampai ke 3 pada gambar kemudian nilai tersebut di turunkan brdasarkan baris pada data set hal itu dapat dilihat pada baris ke 4 sampai ke 9 pada gambar kemudian selanjutnya mengimport library numpy yang berguna untuk oprasi vektor dan matrix karna data diatas berupa data matrix maka library ini di gunakan.

```

1 # shuffle rows
2 muaraenim = muaraenim.sample(frac=1)
3 # split training and testing data
4 muaraenim_train = muaraenim[:500]
5 muaraenim_test = muaraenim[500:]
6 muaraenim_train_att = muaraenim_train.drop(['pass'], axis=1)
7 muaraenim_train_pass = muaraenim_train['pass']
8 muaraenim_test_att = muaraenim_test.drop(['pass'], axis=1)
9 muaraenim_test_pass = muaraenim_test['pass']
10 muaraenim_att = muaraenim.drop(['pass'], axis=1)
11 muaraenim_pass = muaraenim['pass']
12 # number of passing students in whole dataset:
13 import numpy as np
14 print("Passing: %d out of %d (%.2f%%)" % (np.sum(
15     muaraenim_pass), len(muaraenim_pass), 100*float(np.sum(
16     muaraenim_pass)) / len(muaraenim_pass)))

```

- selanjutnya yaitu membuat pohon keputusan pada baris pertama yairu memasukan librari tree kemudian pada baris kedua dibuat variabel palem-bang dengan nilai DecisionTreeClasifier yang merupakan paket atau fungsi dari scikit-learn yang merupakan class yang mampu melakukan multi



Gambar 2.13 hasil

class. sedangkan `max_depth=5` merupakan untuk penyesuaian data terhadap pohon keputusan itu sendiri.

```
1 # fit a decision tree
2 from sklearn import tree
3 palembang = tree.DecisionTreeClassifier(criterion="entropy",
4     max_depth=5)
5 palembang = palembang.fit(muaraeni, _train_att,
6     muaraeni_train_pass)
```



Gambar 2.14 hasil

- selanjutnya yaitu pembuatan gambar dari pohon keputusan yang tadi di buta pada codingan sebelumnya pada baris pertama codingan ya itu mengimport library `graphviz` kemudian pada baris ke dua yaitu pemberian nilai pada variabel baru `dot_data` nilainya diambil dari pembuatan pohon keputusan tadi kemudian di tentukannya nilai benar dan salah dari codingan tersebut setelah itu dibuatlah sebuah variabel untuk menampung hasil eksekusi tersebut kemudian variabel tersebut di running.

```
1 # visualize tree
2 import graphviz
3 dot_data = tree.export_graphviz(palembang, out_file=None,
4     label="all", impurity=False, proportion=True,
5     feature_names=list(muaraenim_train_att), class_names=["fail", "pass"],
6     filled=True, rounded=True)
7 graph = graphviz.Source(dot_data)
```

- selanjutnya pembuatasn method untuk menyimpan data pohon dengan menarik data langsung dari pohon keputusan.

```
1 # save tree
2 tree.export_graphviz(palembang, out_file="student-performance
3     .dot", label="all", impurity=False, proportion=True,
4     feature_names=list(muaraenim_train_att),
5     class_names=["fail", "pass"], filled=True, rounded=True)
```

```

1 digraph TD
2   node [shape=box, style="filled, rounded", color="black", fontname=helvetica] ;
3   edge [fontname=helvetica] ;
4   0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dcc9"] ;
5   1 [label="schoolsup_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdf6f0"] ;
6   0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
7   2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"] ;
8   1 -> 2 ;
9   3 [label="reason_reputation <= 0.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"] ;
10  2 -> 3 ;
11  4 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
12  3 -> 4 ;
13  5 [label="paid_no <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"] ;
14  4 -> 5 ;
15  6 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="ffffff"] ;
16  5 -> 6 ;
17  7 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
18  6 -> 7 ;
19  8 [label="health <= 2.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"] ;
20  7 -> 8 ;

```

Gambar 2.15 hasil

```

1 digraph TD
2   node [shape=box, style="filled, rounded", color="black", fontname=helvetica] ;
3   edge [fontname=helvetica] ;
4   0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dcc9"] ;
5   1 [label="schoolsup_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdf6f0"] ;
6   0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
7   2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"] ;
8   1 -> 2 ;
9   3 [label="reason_reputation <= 0.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"] ;
10  2 -> 3 ;
11  4 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
12  3 -> 4 ;
13  5 [label="paid_no <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"] ;
14  4 -> 5 ;
15  6 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="ffffff"] ;
16  5 -> 6 ;
17  7 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
18  6 -> 7 ;
19  8 [label="health <= 2.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"] ;
20  7 -> 8 ;

```

Gambar 2.16 hasil

8. selanjutnya pada codingan berikut yaitu digunakan untuk mencetak nilai score rata-rata dari ketepatan data yang telah di olah.

```
1 palembang.score(muaraenim_test_att, muaraenim_test_pass)
```

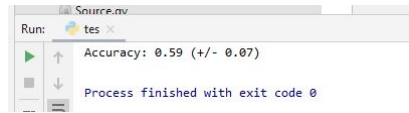
9. selanjutnya yaitu digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut, pada codingan tersebut pada baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel palembang setelah hal tersebut dilakukan kemudian data tersebut di eksekusi.

```
1 from sklearn.model_selection import cross_val_score
```

```

2 scores = cross_val_score(palembang, muaraanim_att,
    muaraanim_pass, cv=5)
3 # show average score and +/- two standard deviations away
4 #(covering 95% of scores)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
    std() * 2))

```



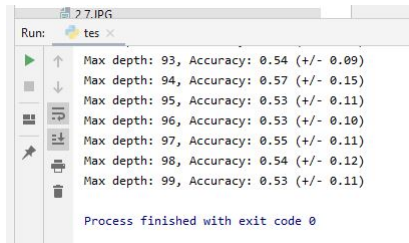
Gambar 2.17 hasil

10. membuat rank akurasi dari 1 sampai 100 untuk melihat akurasi data apakah datatersebut terdapat di rata rata 60 persen atau tidak dengan cara membuat lagi variabel baru dengan nilai tree diadalamnya jadi hampir mirip seperti membuat pohon keputusan namun ini langsung dalam bentuk rata rata akurasi yanlebih spesifik.

```

1 for max_depth in range(1, 100):
2     palembang = tree.DecisionTreeClassifier(criterion="
    entropy", max_depth=max_depth)
3     scores = cross_val_score(palembang, muaraanim_att,
    muaraanim_pass, cv=5)
4     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
    max_depth, scores.mean(), scores.std() * 2))

```



Gambar 2.18 hasil

11. untuk selanjutnya yaitu menentukan nilai untuk grafik hampirsama dengan nilai akurasi tadi pertama tentukan rank atau batasan nilai itu disini batasannya di mulai dari 1 sampai 20 dengan menggunakan nilai tree tadi maka hasilnya dapat ditentukan kemudian buat variabel i untuk penomoran tiap record yang keluar atau recod hadil dari eksekusi tree tersebut.

```

1 depth_acc = np.empty((19,3), float)
2 i = 0
3 for max_depth in range(1, 20):
4     palembang = tree.DecisionTreeClassifier(criterion="
    entropy", max_depth=max_depth)

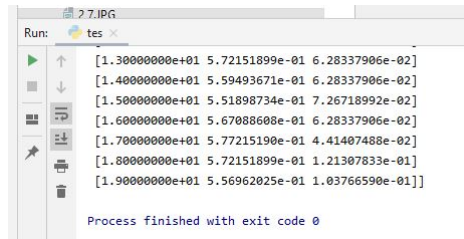
```



```

5     scores = cross_val_score(palembang, muaraanim_att,
6                               muaraanim_pass, cv=5)
7     depth_acc[i,0] = max_depth
8     depth_acc[i,1] = scores.mean()
9     depth_acc[i,2] = scores.std() * 2
10    i += 1
11 print(depth_acc)

```



Gambar 2.19 hasil

- terakhir yaitu pembuatan grafik untuk pembuatan grafik diambil data dari codingan sebelumnya yang 20 record tadi dengan cara mengimport libray matplotlib.pyplot yang di inisialisasi menjadi bakwankemudian inisialisasi tersebut di eksekusi.

```

1 import matplotlib.pyplot as plt
2 fig, ax = plt.subplots()
3 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
4            [:,2])
5 plt.show()

```

### 2.1.3 Penanganan Error

- skrintut error
- kode error dan jenis errornya .

```

import graphviz
dot_data = tree.export_graphviz(palembang, out_file=None, label="all
                                feature_names=list(muaraanim_train_a
                                filled=True, rounded=True)

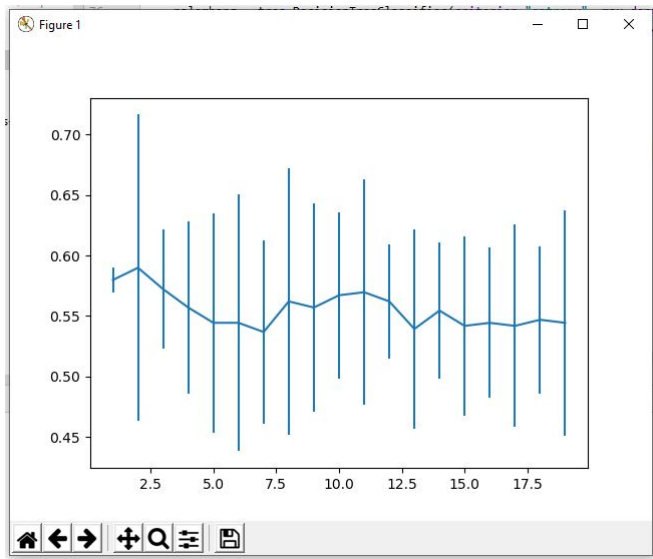
graph = graphviz.Source(dot_data)
graph

```

pada codingan tersebut error karena graphiznya belu di install

- Solusi pemecahan masalah

buka CMD komputer anda run as administrator koemudian masukan perintah conda install graphviz.

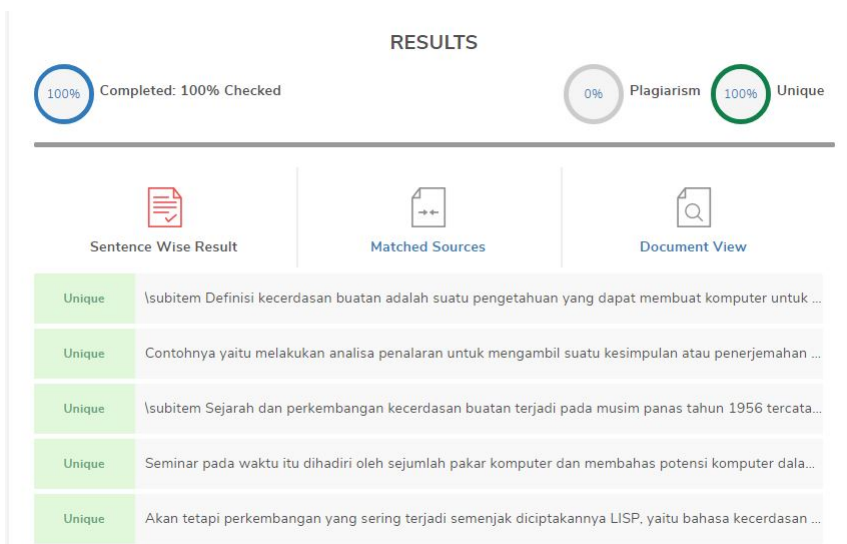


Gambar 2.20 hasil

```
Run: tes
Traceback (most recent call last):
  File "D:/NAJIB/SEMESTER 6_NAJIB/AI/Chapter2/tes.py", line 2, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
Process finished with exit code 1
```

Gambar 2.21 Error

## 2.1.4 Plagiat



**Gambar 2.22** Error

## BAB 3

---

## CHAPTER 3

---



## BAB 4

---

## CHAPTER 4

---



## BAB 5

---

## CHAPTER 5

---





## BAB 6

---

## CHAPTER 6

---



## BAB 7

---

## CHAPTER 7

---

